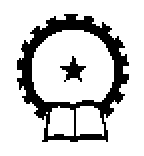
MATLAB 7.0 应用集锦

林雪松　周　婧　林德新　编著



机 械 工 业 出 版 社

本书是为初、中级计算机用户迅速掌握 MATLAB 的应用而编写的。

MathWorks 公司自从 1984 年推出 MATLAB1.0 版以来，经过不断完善，目前已形成相对稳定的版本 MATLAB7.0 版。它比以前的版本无论在桌面布局、数学运算能力、图形绘制的方便性和程序的调试/排错方面都有很大的改进。它帮助你从繁琐的数学运算和图形绘制的手工劳动中解放出来，使数学运算和图形绘制走向自动化，并且通过科学计算使生产与管理工作的效益大为提高。

本书不是繁琐地介绍 MATLAB 软件中对指令、语句、函数和使用条件，而是通过大量实例来说明指令、语句、函数的应用和解题方法，并介绍某些函数的产生、编程和计算方法，使读者容易理解和记忆。

本书的内容由浅入深，例题内容涉及数学、工程设计最优化、规划设计、金融分析、电工计算和自动控制系统的分析等，并附有游戏。每道例题都经过筛选，例题中所引用的指令、函数和程序大多附有注解予以说明，并通过上机调试。

本书适用于从事数学计算、工程设计、规划设计、自动控制以及其他从事数学计算的专业人士，也可以作为大专院校理工科学生的辅助读物。

图书在版编目（CIP）数据

MATLAB 7.0 应用集锦/林雪松等编著 .—北京：机械工业出版社，2005.9
ISBN 7-111-17345-7

Ⅰ.M… Ⅱ.林… Ⅲ.计算机辅助计算－软件包，MATLAB 7.0－程序设计 Ⅳ.TP391.75

中国版本图书馆 CIP 数据核字（2005）第 102143 号

机械工业出版社（北京市百万庄大街 22 号 邮政编码 100037）
策划编辑：孙流芳 责任编辑：罗 莉
版式设计：霍永明 责任校对：王 欣
封面设计：马精明 责任印制：
印刷厂印刷
2006 年 1 月第 1 版第 1 次印刷
787mm×1092mm 1/16·27.75 印张·686 千字
0001—4000 册
定价： 元

凡购本书，如有缺页、倒页、脱页，由本社发行部调换
本社购书热线电话（010）68326294
封面无防伪标均为盗版

前　言

MATLAB作为一个优秀的数学软件，目前被广泛应用于各个领域，例如科学计算、工程设计、自动控制、企业管理、统计分析和金融分析等。

MATLAB所以得到广泛的应用，这是与其所具有的特色分不开的。首先它把变量看作数组，以数组作为运算单元。当数组是一维1元素时它就是一个标量。当数组为一维时，它就是一个向量。当数组是二维时，它就是一个矩阵。MATLAB在变量运算中无须为变量定维，它会根据运算范围自动定维，所以使用十分方便。

其次，MATLAB的编程像通常书写数学公式那样方便，无须编译和连接。只要会书写数学公式，基本上就会编程。它把常用的、专业应用的程序列入函数库，当需用时，随时可以调用。对于MATLAB7.0版本，它具有超过1000个数学和工程函数可供访问。若对于函数库某个函数的使用规则和语法不熟悉，只要键入“help 函数名”，或者“doc 函数名”即可得到帮助。

再次，MATLAB的可视化功能，它能执行数据、函数的绘图功能，能根据要求绘制二维、三维的数据图、函数图和专业用图形，使数据和函数不再是抽象、枯燥的东西。

MATLAB自从1984年推出1.0版以来，经过不断发展，已日臻完善。目前已发展成相对稳定的7.0版。新版本在桌面设置的多样化，文件查询的便利，数学函数的扩充与改进，丰富的数据可视化功能，数据输入与输出的方便性，帮助文本的内容丰富，以及它能发展与其他语言，如Excel、C/C++、Java、Web、Visual Basic COM的综合应用。

此外，MATLAB有30多个工具箱，例如生物仪器工具箱、通信工具箱、控制工具箱、曲线拟合工具箱、数据库工具箱、数据取得工具箱、滤波器设计工具箱、金融工具箱、金融衍生工具箱、金融时间序列工具箱、模糊逻辑工具箱、最优化工具箱、信号处理工具箱、图像取得工具箱、图像处理工具箱、符号数学工具箱、小波工具箱、鲁棒控制工具箱、神经网络工具箱、偏微分方程式工具箱、线性矩阵不等式、绘制地图（LMI）工具箱（Mapping）工具箱和统计分析工具箱等。在每一个工具箱中，读者都能找到它的帮助文本、演示和产品介绍。

目前与 MATLAB 相关的数学软件不下数十种，国内常用的有以下几种:

(1) MATHMATICS——它的结构比较严谨，但使用范围不及 MATLAB。

(2) MAPLE——它是以符号数学为基础的数学软件，MATLAB 可以连接它的函数库。

(3) LINGO——它是以处理中、大规模的线性规划问题为对象的数学软件。

本书是一本 MATLAB 的入门书，作者想通过本书引起读者对它的兴趣，以便使它在各行各业的应用中发挥作用，从而提高生产和管理的效益。本书第 1 ~ 6 章介绍 MATLAB 基础知识。第 7 ~ 12 章介绍 MATLAB 在数学、工程、管理、自动控制和金融中的应用。在第 2 ~ 12 章的末尾附有习题供自我测试，以巩固学习成果。本书有关 MATLAB 的例题有 300 余个，内容由浅入深，例题涉及数学、物理、电气、财经和自动控制。每道例题中所引用的指令和函数大多附有注解予以说明，并都通过上机调试。

由于 MATLAB 内容极为丰富，本书显然不可能包含 MATLAB 的所有应用，所以只能起到抛砖引玉的作用。

本书在编著过程中的初稿经上海理工大学徐国华教授、深圳大学宋富高副教授审阅，也得到孙流芳编审的鼓励，他们提供了宝贵的意见，特此表示感谢。

由于编者的水平有限，写作过程中错误难免，希望读者批评指正，并提出宝贵意见。E-mail 地址: 林雪松 alexlin888@yahoo.com; 周婧 jeanzhou@acnielsen.com.cn; 林德新 linx2358@yahoo.com.cn。

作 者

2005 年 8 月

目　录

第 1 章　概　　述

1.1　什么是 MATLAB

MATLAB 是一种高性能的、用于科学和技术计算的计算机语言。它使得计算和图像一体化，编程容易，它所使用的环境，将问题和解答用大家熟悉的数学标记来表达。自从 1984 年美国 MathWorks 公司首先推出 MATLAB 1.0 版以来，受到科学技术界的广泛欢迎。后来 MathWorks 公司不断更新和充实 MATLAB。与 MATLAB 组合在一起的软件是 SIMULINK，它是用来对动态系统进行建模、仿真和分析的软件，它支持连续、离散系统和非线性系统。SIMULINK 不能独立运行，只能在 MATLAB 环境中运行。MATLAB 的版本发展年表大致见表 1-1。

表 1-1　MATLAB 版本发展年表

日期/年，月	MATLAB 版本	SIMULINK 版本
1984	1.0	SUMULAB①
1992	4.0	1.3
1997	5.0	2.0
2001	6.1	4.0
2003	6.5	5.1
2004.7	7.0	6.0

① SIMULINK 的前期产品。

目前已发布至 MATLAB 7.0 版。MATLAB 的全名是 Matrix Laboratory，意思是矩阵实验室。早期的 MATLAB 是建立在 DOS 操作系统上，到了 20 世纪 90 年代才发展成在 Windows 操作系统上，它的功能也大为加强，不但能从事数值计算，还具有从事符号解析运算、逻辑运算、数理统计、控制系统分析、最优化运算、金融的分析、数据的可视化、动力系统的建模和仿真等功能。MATLAB 具有以下特点：

1. 数学和计算

MATLAB 能执行矩阵运算、符号运算、公式化简、线性与非线性方程式求解、高阶方程求根、线性规划、数理统计、微分和积分运算、微分方程求解、最优化运算以及自动控制系统的分析计算等。MATLAB 可以说改变了计算数学的历史，使得复杂的计算变得容易，使脑力劳动者从大量繁琐的计算中解放出来，使数学分析和计算成为轻松愉快而有意义的事情。例如，过去分析高阶、多变量自动控制系统的稳定性和参数选择往往需要数天的时间，而采用 MATLAB 以后，只需要若干分钟即可完成，大大缩短了分析和设计时间，并且还能提供详尽的图表和过渡过程曲线。

2. 直译式的编程语言

MATLAB 既可以在命令窗口直接进行运算，也可以在 M 文件窗口中进行编程运算，而无须编译（compile）和链接（link）。由于 MATLAB 包含 1000 多条数学函数和工程计算函

数，因而可直接调用而不用另行编程，例如计算逆矩阵 A，只要调用 inv（A）即可执行，所以大大节省编程时间和运算时间。

3. 先进的数据可视化功能

MATLAB 能够按数据产生高质量的二维和三维的数据图形，如散点图、直方图、饼图、树干图、阶梯图、向量图和函数图形等，给用户提供既直观而又精确的图像。

4. 丰富的工具箱

为协助解决各个技术领域的应用，MATLAB 设置了 30 多个工具箱。例如生物仪器工具箱、通信工具箱、控制工具箱、鲁棒控制工具箱、模糊逻辑工具箱、滤波器设计工具箱、曲线拟合工具箱、神经网络工具箱、最优化工具箱、金融工具箱、金融衍生工具箱、金融时间序列工具箱、系统识别工具箱、统计工具箱、偏微分方程工具箱和图像处理工具箱等。读者可从工具箱中找到相应帮助文本、应用举例和演示程序（DEMO）。

1.2 MATLAB7.0 的安装

MATLAB7.0 对 PC 的要求：

(1) CPU　Pentium III、IV、Xeon，Pentium M，AMD Athlon，Athlon XP，Athlon MP

(2) 操作系统　Windows 2000/XP/NT 4.0 或其他

(3) 显卡　16MB 以上

(4) 显示器　支持 256 色，分辨率为 800×600pi

(5) 内存　256MB 以上，建议 512MB

(6) 硬盘　1GB 以上

MATLAB7.0 的安装：将 MATLAB7.0 的第一张光盘插入CD－ROM 中，则自动进行预安装，预安装结束将显示 MATLAB7.0 的画面，接着转到欢迎进入 MathWork 安装，并显示本程序将安装 MATLAB 家族产品的对话框，如图 1-1 所示。该对话框有两个选项，安装或用新的 PLP（个人许可密码）和更新许可不需任何安装。并说明该软件是受版权法保护，在安装进程中，必须同意按照软件许可协议所规定的使用限制。任何未经授权使用、复制，将受到法律起诉。

点击“Next”按钮，则显示输入个人姓名、公司名称和个人许可密码（PLP），PLP 是 5 位数字一组的 10 组数字，开头二位数字则是版本号。如图 1-2 所示，如果没有 PLP，则可连接 Internet，按“Get my PLP”按钮向 MATHWORKS.COM 网站或国内代理联系。

点击“Next”按钮，则显示许可协议，如图 1-3 所示，点击“Yes”单选按钮，则表示同意。

点击“Next”按钮，则显示选择安装画面，如图 1-4 所示。用户可选择典型安装或指定安装内容。

点击“Next”按钮，则进入选择目标文件夹为下一步安装，如图 1-5 所示。

点击“Next”按钮，则显示安装内容供确认，如图 1-6 所示。

点击“Install”按钮后则开始安装，并显示完成安装进度百分数，如图 1-7 所示。

当第一张光盘装完后，安装进度约大于 50%，屏幕提示请插入第二张光盘，以装入系统说明文件。当安装结束，则点击“Finish”按钮即可。

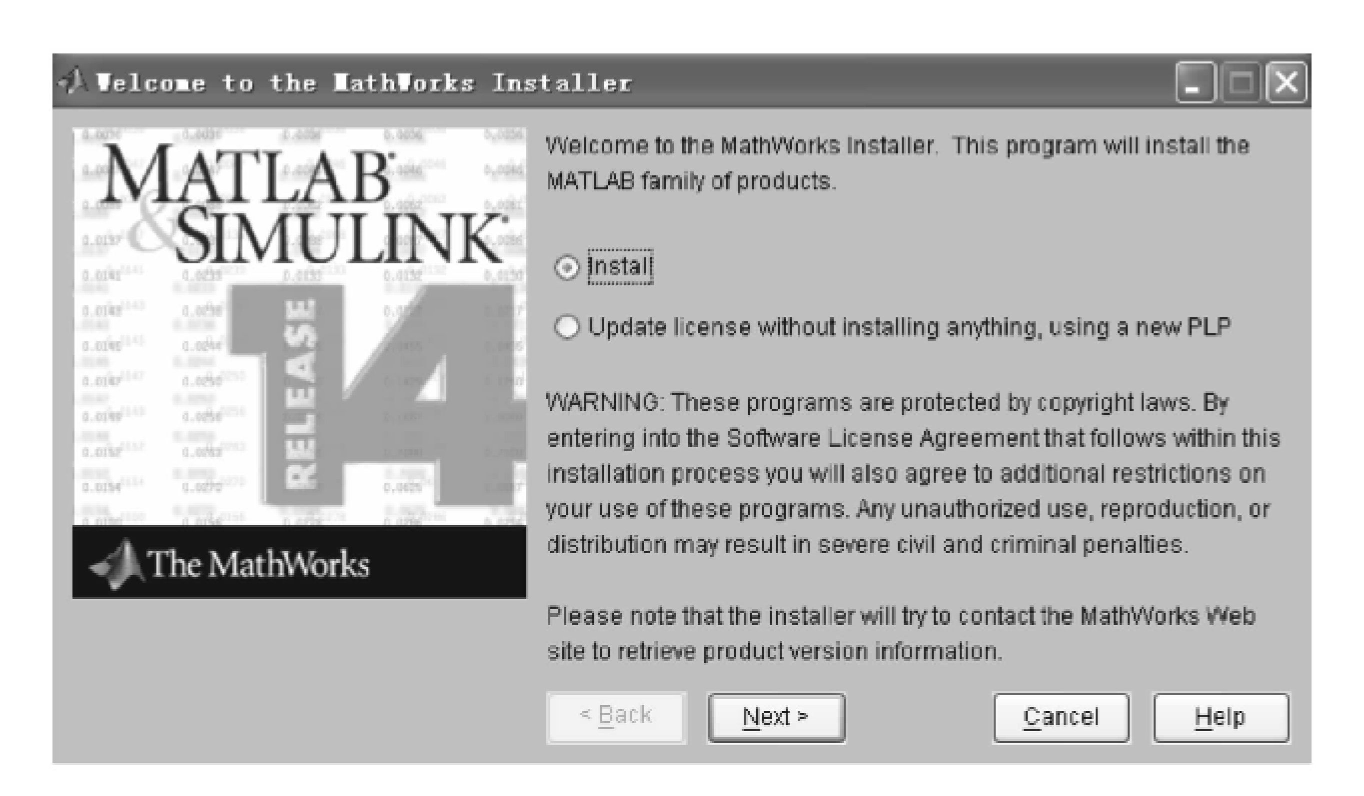


图 1-1　安装 MATLAB7.0

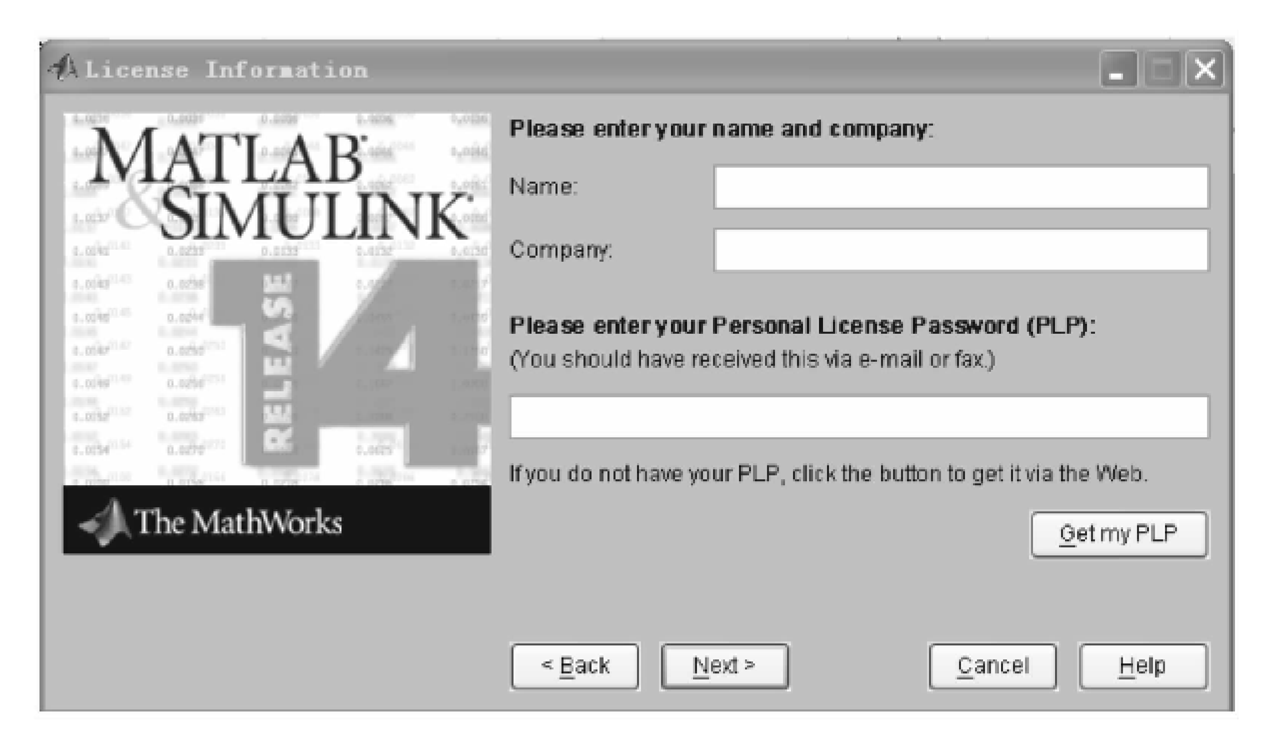


图 1-2　输入姓名、公司名称和个人许可密码

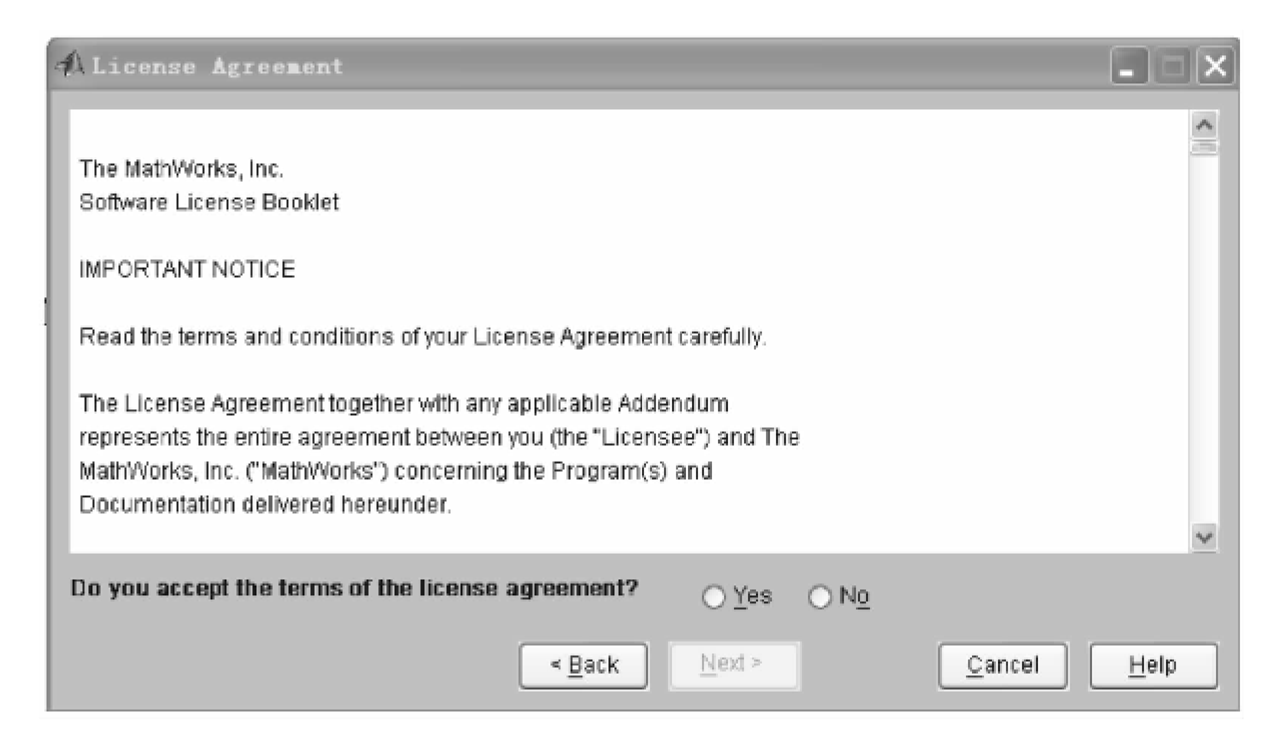


图 1-3 许可协议

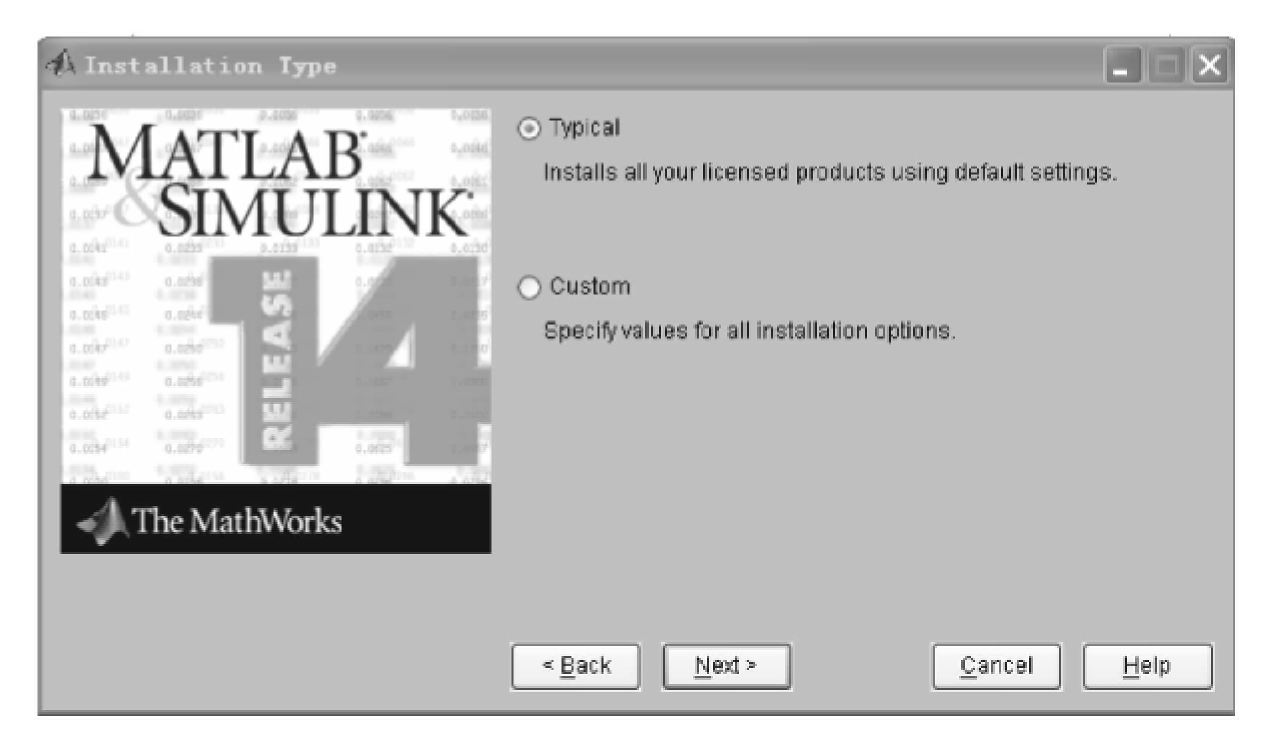


图 1-4 选择安装内容

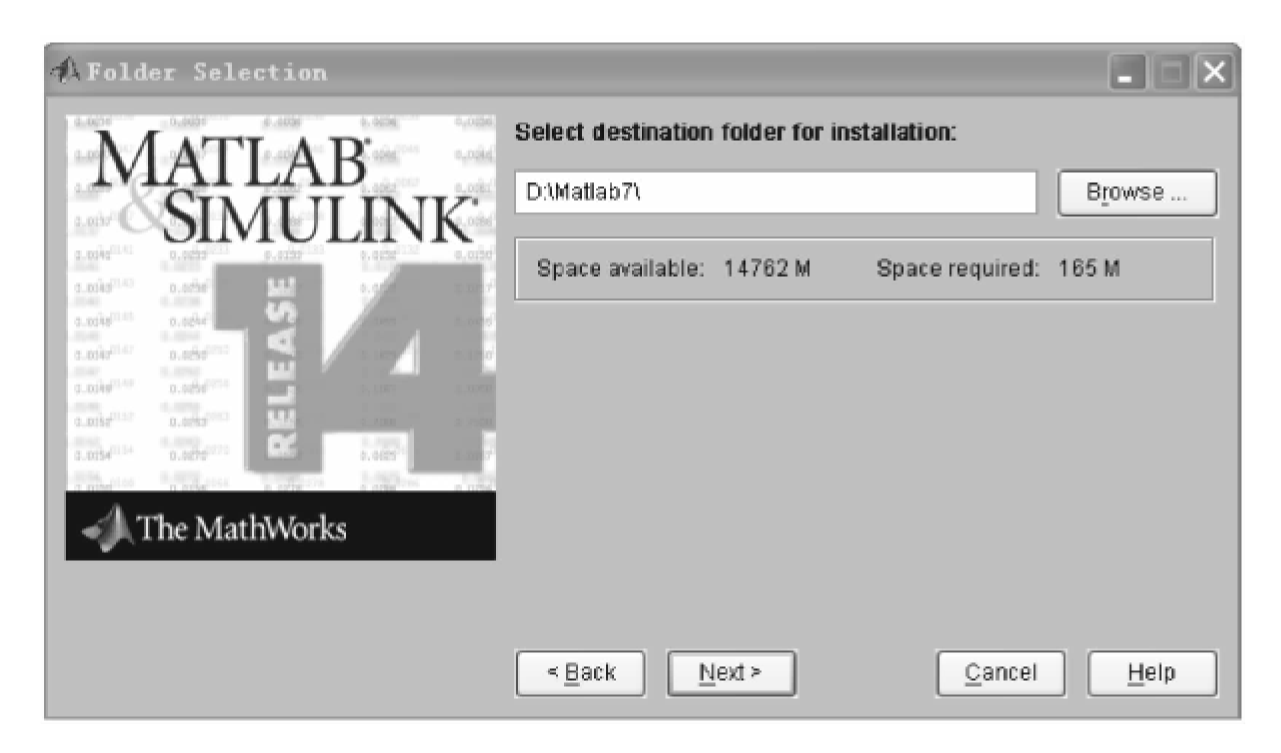


图 1-5　选择目标文件夹为下一步安装

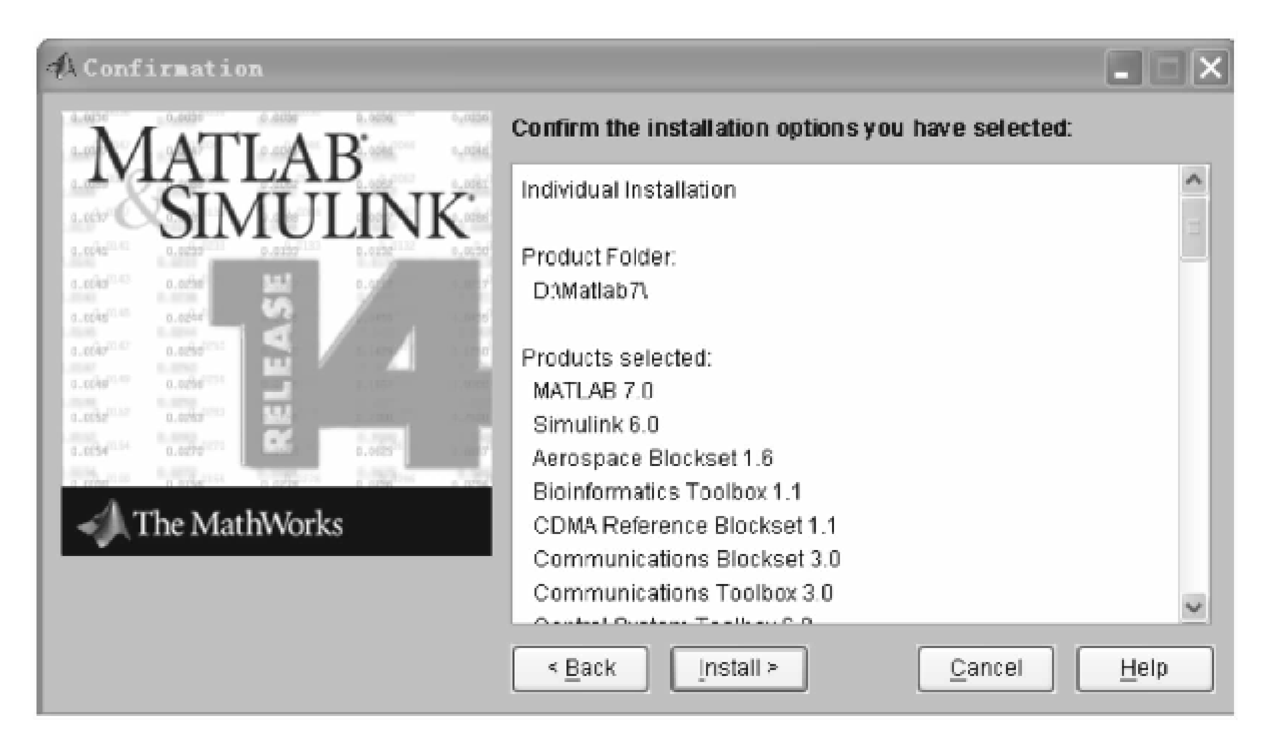


图 1-6　确认安装内容

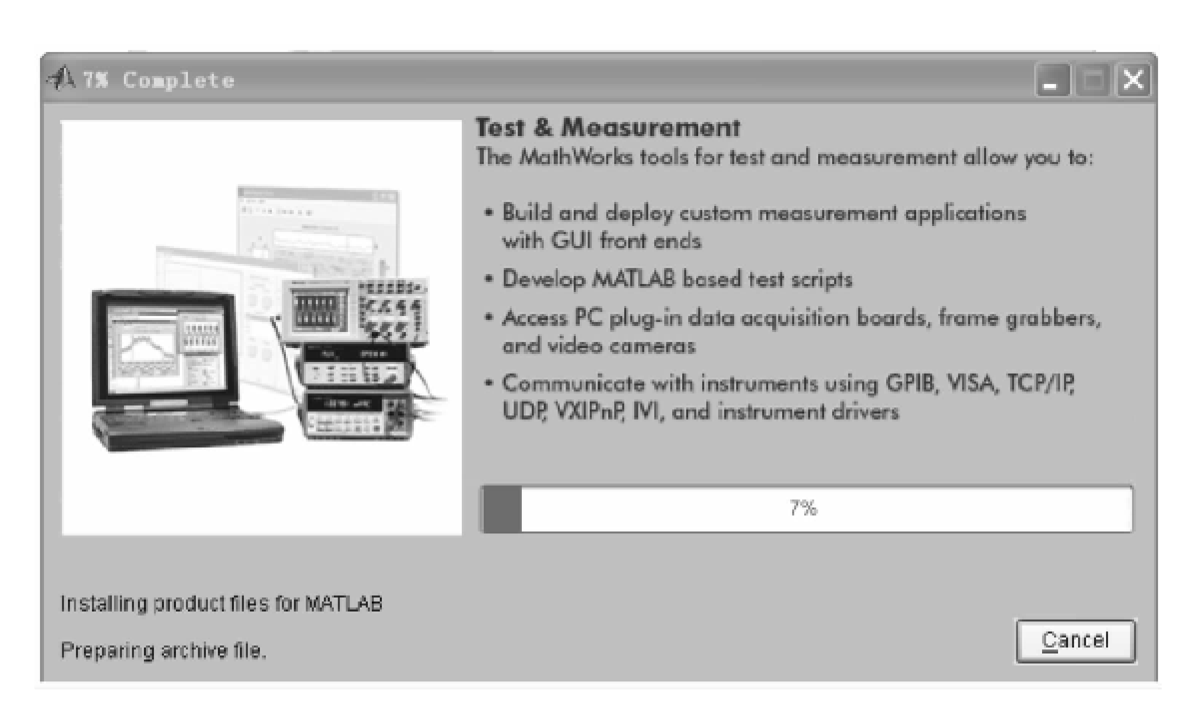


图 1-7　显示安装进度

1.3　MATLAB7.0 桌面的概貌

MATLAB 启动后的默认桌面如图 1-8 所示。它有以下几个窗口。

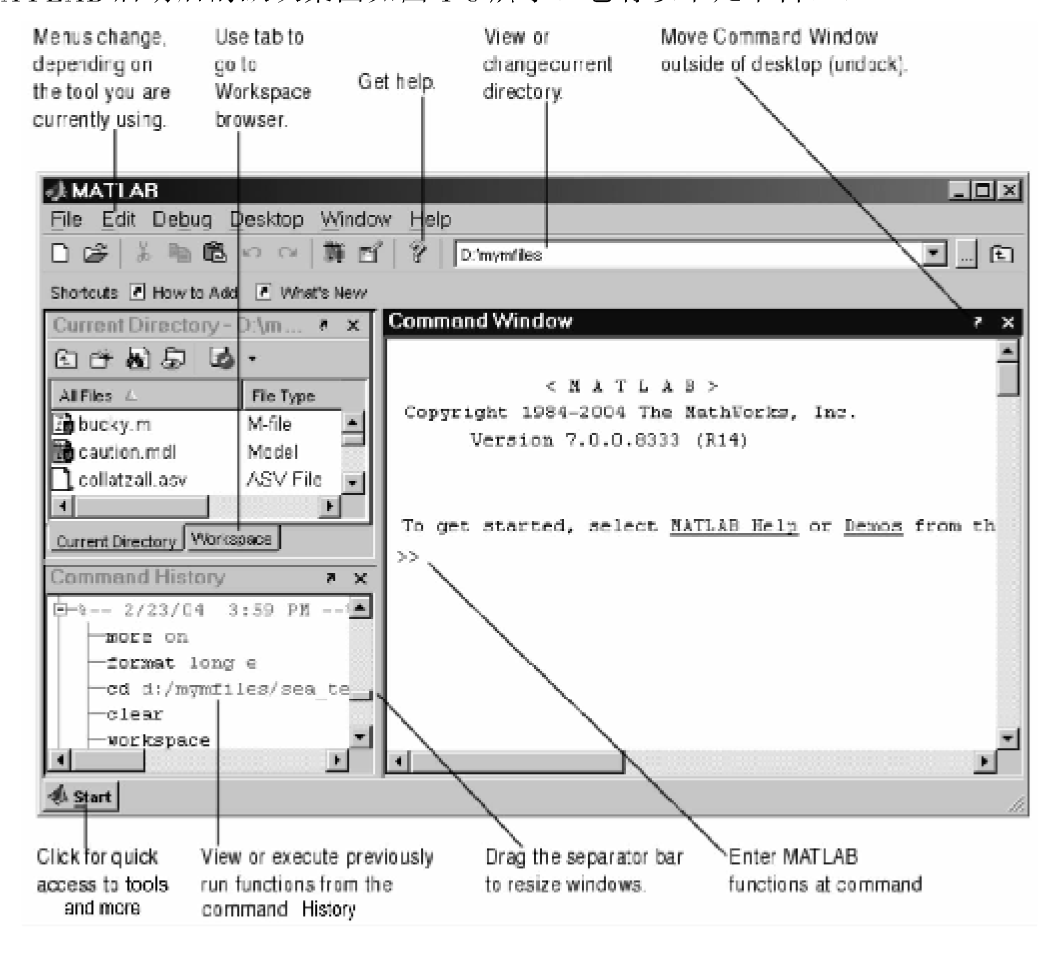


图 1-8　MATLAB7.0 的默认桌面布局

1. 命令窗口（Command Window）

它处于窗口的右侧，用来输入数据、操作命令和显示运行结果。

2. 当前目录（Current Directory）

它和工作空间（Workspace）浏览器共用一个窗口，处于主窗口的左上方。在该窗口下方，点击 Current Directory 框或 Workspace 框，可以进行两个窗口的切换。Current Directory 用来显示当前目录下的所有文件名和文件类型。Workspace 则用来显示已经使用过的变量名、变量的类型和变量的大小。

3. 命令历史（Command History）窗口

它处于当前目录和工作空间窗口之下，用来记录已经使用过的命令、函数或数据和使用时间以便日后查找。当需要将已经用过的命令、函数或数据重复调入命令窗口使用时，只要双击该命令，即可自动将它们调入命令窗口中执行。这对使用者来说是十分方便的，可以省去重复键入。当需要清除命令历史窗口时，则可用右键点击历史窗口，则显示选择框，从选择框中选取 Clear Entire History 即可。

4. Start 窗口

它位于主窗口的左下角，具有多个子菜单，如 Matlab、Toolboxes、Simulink、Blockset、Shortcuts、Desktop Tools、Web、Preference、Find files、Help 和 Demo 等。点击它，就能实现对子菜单的快速访问。

5. 数组编辑器（Array Editor）窗口

它是用来显示或编辑数组的窗口，也是 MATLAB7.0 新增的功能。数组编辑器的打开，可以通过以下 3 种方式：

(1) 在工作空间浏览器中，用鼠标左键双击变量名。

(2) 在命令窗口使用 openvar（“变量名”）。

(3) 在工作空间浏览器中，选中变量，随后点击 open selection 图标。

数组编辑器的窗口如图 1-9 所示。

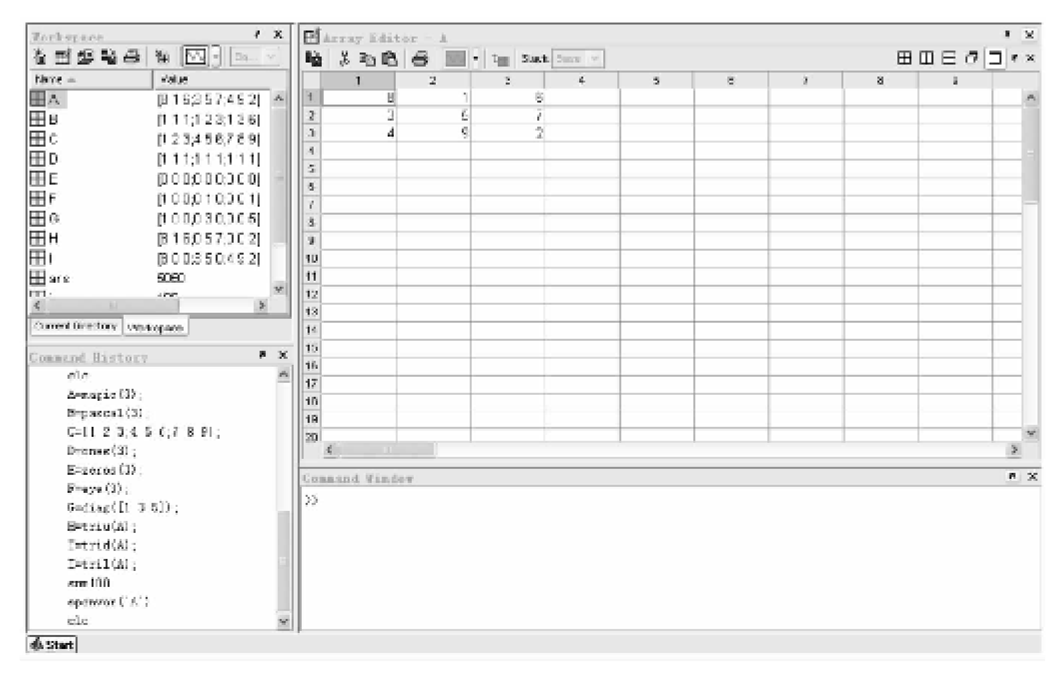


图 1-9　工作空间浏览器中的数组编辑器

利用数组编辑器右侧工具栏的注释为 Tile 图标（用作矩阵块设置），可以查看多个变量，如矩阵 ***A***、***B***、***C***、***D***、***E***、***F*** 的内容，如图 1-10 所示。其中 ***A*** 为 3 阶魔方矩阵，***B*** 为 3 阶 Pascal 矩阵，***C*** 为顺序数矩阵，***D*** 为全 1 矩阵，***E*** 为零矩阵，***F*** 为单位矩阵。

数组编辑器还有修改数据、剪切、复制、粘贴和删除的功能（使用数组编辑器中工具栏左侧的工具图标）。使用工具栏右侧工具图标，可以对变量进行直排、横排和块排列；对命令窗口数据进行交换以及对 Excel 软件数据进行交换的功能。

A	1	2	3	4
1	8	1	6	
2	3	5	7	
3	4	9	2	
4				

B	1	2	3	4
1	1	1	1	
2	1	2	3	
3	1	3	6	
4				

C	1	2	3	4
1	1	2	3	
2	4	5	6	
3	7	8	9	
4				

D	1	2	3	4
1	1	1	1	
2	1	1	1	
3	1	1	1	
4				

E	1	2	3	4
1	0	0	0	
2	0	0	0	
3	0	0	0	
4				

F	1	2	3	4
1	1	0	0	
2	0	1	0	
3	0	0	1	
4				

图 1-10　用数组编辑器显示 6 个矩阵

6．M 文件编辑器（M-files Editor）窗口

由点击 file/new/m-file 产生，也可在命令窗口键入 edit 来实现。M 文件编辑器用来编辑函数 M 文件和文本 M 文件，它是 MATLAB 的程序编制窗口，也即通过编制程序来从事科学计算。

命令窗口是用来输入命令或命令集合，并立即执行，显示运算结果或自动反馈信息（如 Help 信息、操作出错信息等）。它也是一个操作者与 MATLAB 交互的主窗口。在命令窗口上有文件栏（File）、编辑栏（Edit）、查看栏（View）、桌面栏（Desktop）、窗口栏（Window）、帮助栏（Help）等。

在文件栏中有如下几项：

(1) New　　新建 M 文件、图形、模块、用户图形界面
(2) Open　　打开已经保存过的 M 文件、数据文件、图形文件和模块
(3) Import　　导入外部数据文件
(4) Save Workspace As　　另存工作空间文件
(5) Preference　　设置 MATLAB 工作环境参数
(6) Print　　选择打印
(7) Exit　　退出 MATLAB

如果要清除命令窗口数据，可以选择 Edit 菜单，从中选取 Clear Command Window 即可，也可以在命令窗口的提示符下，键入 clc 再回车即可。

如果用户要查找工具箱中最优化工具箱，则可单击主窗口左下角的 Start 按钮，如图 1-11所示。

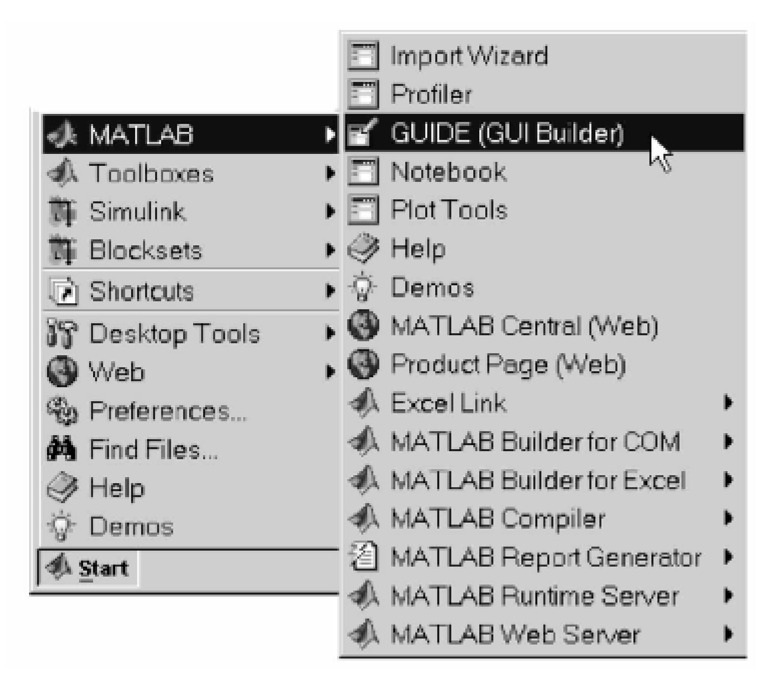


图 1-11　Start 菜单

在子菜单中选取 Toolboxes/More.../Optimization（最优化工具箱）就出现 3 个子目录，即 Help、Demos 和 Product page（Web）。点击 Help，则进入最优化工具箱的帮助文本，如图 1-12 所示。

工作空间浏览器窗口用来显示变量的设置，如图 1-13 所示。它显示变量名称、大小、所占用空间和变量类型。在图 1-13 中，变量 M、im 为数值变量，a、b、c、A、iA 为符号变量。

在图 1-13 中，name 栏为变量名称，value 栏显示变量大小，class 栏显示变量类型。应该指出，变量是由命令窗口设置的，并在工作空间中自动生成，无须用户设定维数。利用工作空间，用户可以很方便地检查用了哪几个变量以及变量的大小与类型，并且可以在工作空间中通过数组编辑器来修改变量矩阵中的各个元素。MATLAB 允许用户用 save 命令保存工作空间中变量列表，使之成为后辍为 MAT 的数据文件，还可以用 load 命令调用已经保存过的数据文件到工作空间浏览器。

在工作空间（Workspace）浏览器窗口标题下有一个工具栏，如图 1-14 所示。

工具栏上各个图标的注释和用途如下：

(1) New variable　　设置新变量

(2) Open selection　　打开数组编辑器

(3) Load date file　　装入数据文件

(4) Save　　保存数据成数据文件

图 1-12　由 Start 菜单进入最优化工具箱的帮助文本

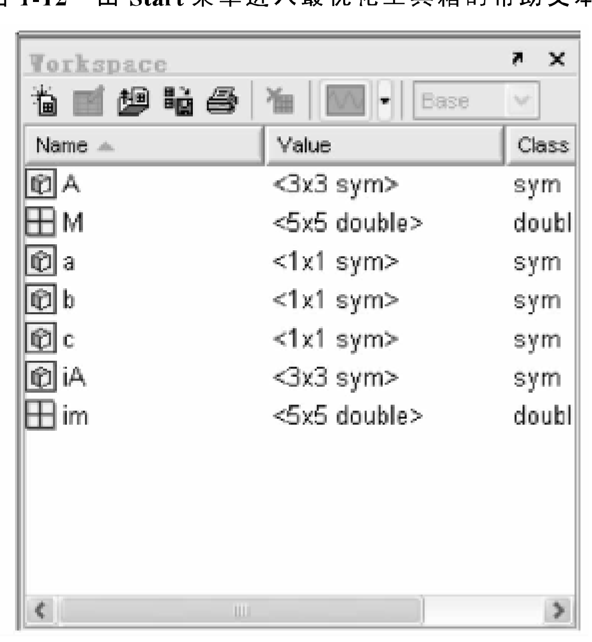


图 1-13　工作空间浏览器窗口



图 1-14　工作空间浏览器窗口下的工具栏

(5) Print　　打印工作空间数据

(6) Delete　　删除变量

(7) Plot all columns　　按照变量的各个列向量，绘制线型图，使数据可视化

(8) Stack　　显示堆栈

要清除工作空间的内容，可以选择 Edit 菜单，从中点击 Clear Workspace 即可，也可以在命令窗口的提示符下，键入 clear 再回车即可。

当前目录窗口，如图 1-15 所示。它用来显示当前是在什么目录下进行工作的，在当前目录窗口下，D: Matlab \Toolbox显示所有文件的名称、文件类型和最后修改日期。用户可以双击文件名来查看文件的内容或执行该文件的程序。当前目录下有一个工具栏如图 1-16 所示。

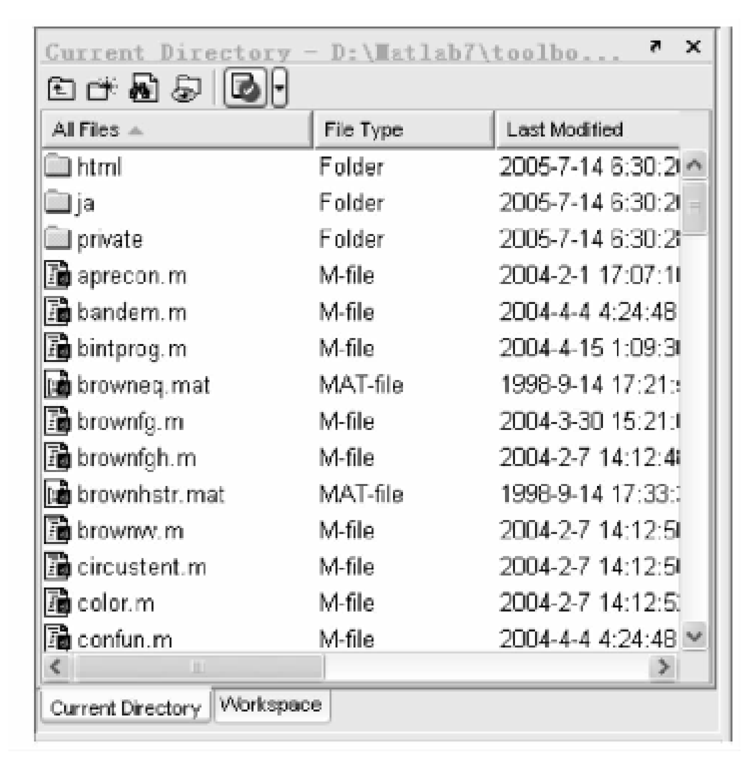


图 1-15　当前目录窗口



图 1-16　当前目录窗口下的工具栏

工具栏上各个图标的注释和用途：

(1) Go to one level　　进入上一级目录

(2) New folder　　建立新文件夹

(3) Find files　　寻找文件

(4) Show visual directory　　显示可视目录

(5) M-lint code chock report　　建立 M 文件编译代码检查报告

(6) Directory report　　显示报告目录

M 文件编辑窗口如图 1-17 所示。在 M 文件编辑器中可以通过编程来执行一系列函数语句和命令，以实现特定的计算任务。函数 M 文件是由内装函数 M 文件（如三角函数、指数函数、行列式求值、逆矩阵计算和线性规划计算函数等）和读者自行编制函数 M 文件组成，详见第 3 章“MATLAB 编程与数据类型”。M 文件的编制通常要经过调试和排除错误。对程序进行编译代码检查，这样有利于改进程序的编制。M 文件的执行，是在命令窗口中进行。在命令窗口中，输入相应的 M 文件名并回车即可。图 1-17 中所示的程序是最简单的程序。它用来计算 1 加到 100 的总和。

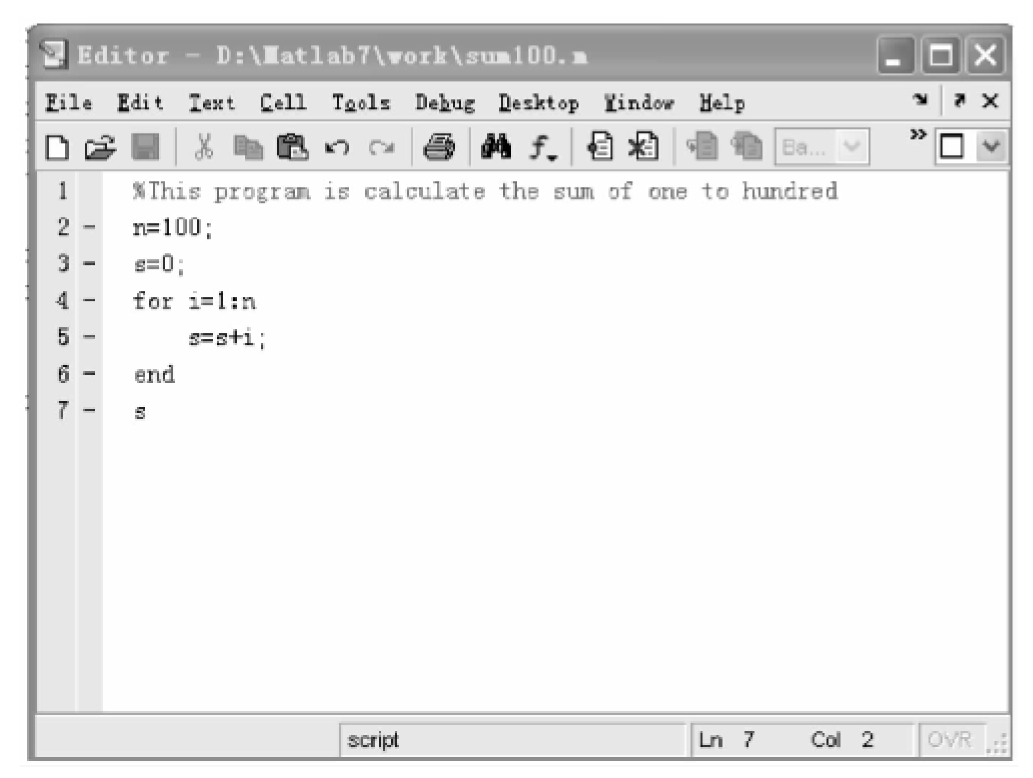


图 1-17　M 文件窗口

1.4　MATLAB7.0 的桌面工具

表 1-2 所列的桌面工具是用来管理 MATLAB 桌面的。虽然不是所有的桌面工具都是在一开始使用时，以默认的形式出现的。如果喜欢用命令行界面，那么也可以用等效的功能，使用桌面工具的特性来实现同样的结果，你可以使用具有相同功能的桌面工具去执行 M 文件的操作。

表 1-2　桌面工具的介绍

桌 面 工 具	说　　明
Array Editor	以表格形式查看数组内容，并且可以编辑它的数值
Command Window	输入数据，运行 MATLAB 函数，执行程序，并显示结果
Command History	查看你在命令窗口输入过函数和命令的运行日志。可以调用在命令历史窗口中的命令，实现在命令窗口的复制、运行及其他
Current Directory Browers	查看文件，执行文件操作，如打开、寻找、查看内容、文件管理和调整文件等
Editor/Debugger	建立编辑，调试 M 文件（包含 MATLAB 函数文件）

（续）

桌面工具	说明
Figures	建立、修改、查看和打印 MATLAB 图形
Help Browser	查看和搜查所有有关 MathWorks 产品的文件
Profiler	用图形界面来改善 M 文件性能
Start Button	运行和访问你所有的 MathWorks 产品中的工具箱和文件，并建立与使用 MATLAB 快捷键
Web Browser	查看与 MATLAB 有关的 HTML 和相关信息
Workspace Browser	查看工作空间中所使用的变量或改变变量的内容

【例 1-1】 编辑两个 M 文件的桌面排列。

图 1-18 所示的桌面排列上，上排显示：用此图标确定文件排列方式，关闭或离开桌面的激活工具，使文件离开桌面工具。左侧显示：Documentbar 用于文件切换，Tabs 用于编辑器切换。

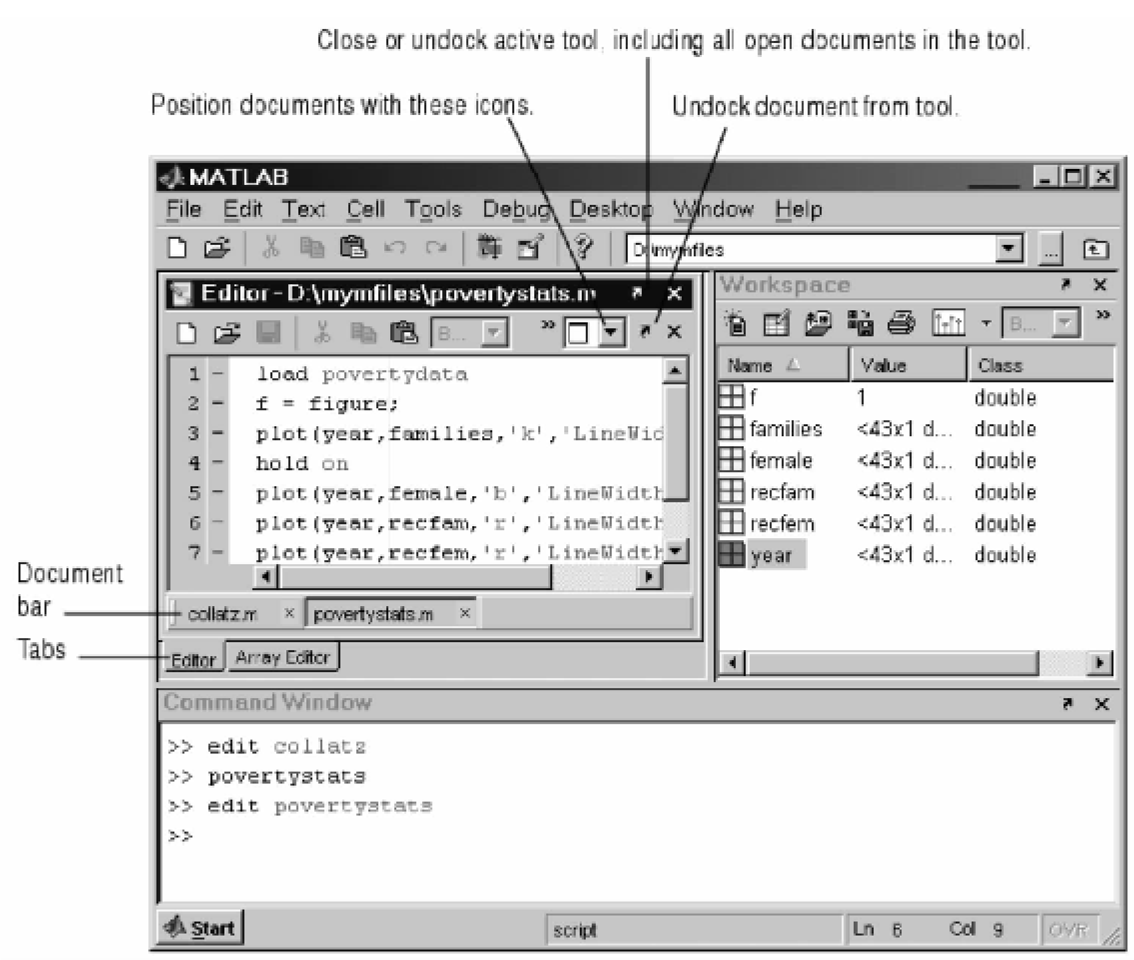


图 1-18 两个 M 文件的桌面排列

【例 1-2】 两个图形引入桌面的例子。

可以在命令窗口输入两个图形的文件名，则图形即显示在命令窗口中。使用图形窗口上的工具栏，可以对图形进行编辑，也可以对图形进行放大、缩小、平移和存储。

图 1-19 所示为图形引入桌面的排列。上排显示：图形引入桌面，文件滑动条隐藏。

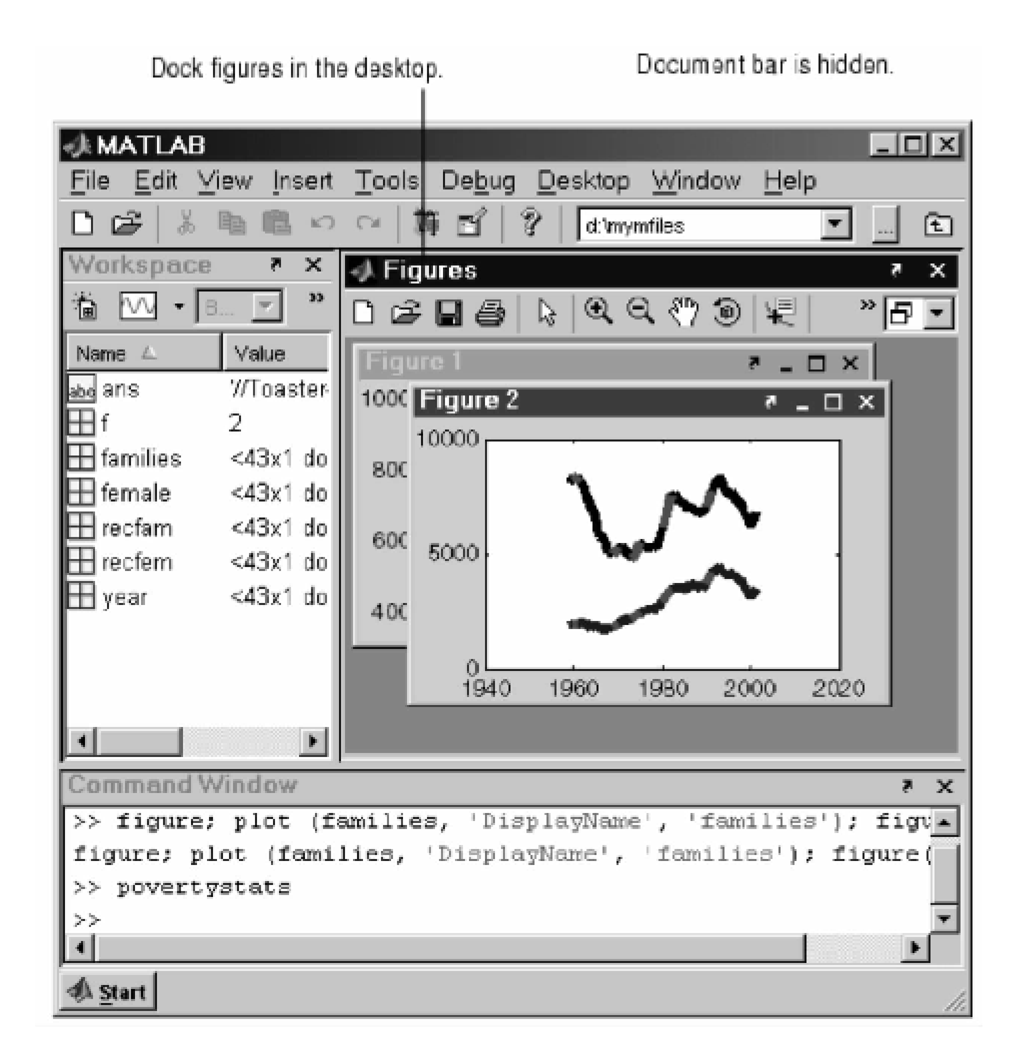


图 1-19　图形引入桌面的排列

当运行结束，MATLAB 可以保存各种形式的桌面布局，使得下次启动 MATLAB 时恢复原来的桌面布局。为了保存桌面布局，可以选择Desktop/save layout菜单即可。

1.5　MATLAB7.0 的新增特性

MATLAB7.0 比以前的版本 6.1 和 6.5 有较大的改进。并增加了新的特性。这些新特性包括以下几方面。这些新特性对于初学者，可暂时不阅读，在阅读第 2、3 章后，再返回阅读这部分内容比较有利。

1. 桌面工具和发展环境

新的桌面提供多种桌面工具，如数组编辑器、命令窗口、命令历史窗口、当前目录浏览器、程序编辑器/调试器、图形显示器、帮助浏览器、启动按钮、网页浏览器和工作空间浏览器等。对于桌面工具的排列可以左右排列、上下排列或分块排列。用户可以对自行设置的桌面布局进行存储，以便下次开机时，重现上一次的桌面布局。

2. 数学运算方面

MATLAB7.0 在数学运算方面比 6.5 版本有 26 项改进。如非双精度数算术，数据类型引入 ones、zeros、eye 和 sum 函数，三角函数中可以用角度来表示，发展了几何问题的计算，1 阶隐式微分方程的求解，改进线性方程式求解，改进了最优化函数和新建数组累加函数等。这里用两个例子予以说明。

【例 1-3】 用 linsolve（A，b）解线性方程组。

以前的版本，对于解线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，其中 $\boldsymbol{A}$ 为系数矩阵，$\boldsymbol{b}$ 为常数项向量。通常用左除法或逆矩阵法求解（详见第 2 章）。在 7.0 版本中则可以用 linsolve 求解。它的解题速度比前者更快。

若 $\boldsymbol{A}$ 为三阶魔方矩阵，$\boldsymbol{b}=\begin{bmatrix}1\\2\\3\end{bmatrix}$则在命令窗口中输入如下程序：

```
>>A = magic(3)
A =
     8     1     6
     3     5     7
     4     9     2
>>b = [1;2;3];
>>x = linsolve(A,b)
x =
     0.0500
     0.3000
     0.0500
```

【例 1-4】 累加数组函数 accumarray。

它的书写格式为

A = accumarray(ind，val)

共中，ind 为索引数组；val 为行向量。val 的列数必须与 ind 的行数相等。val 的元素值是以索引数组的下标写入矩阵 A。它常用于创建稀疏矩阵。

例如若ind = [1 5 4 3 2;1 2 3 4 5];val = [2 3 5 8 11];在命令窗口输入如下程序：

```
>>ind = [1 1;5 2;4 3;3 4;2 5];          %索引数组
>>val = [2 3 5 8 11];                   %元素值向量
>>A = accumarray(ind,val)               %累加数组
A =
     2     0     0     0     0
     0     0     0     0    11
     0     0     0     8     0
     0     0     5     0     0
     0     3     0     0     0
```

若 ind = [1 2;3 2;5 5;5 5];val = [10.1 10.2 10.3 10.4];则在命令窗口输入如下程序：

```
>>ind = [1 2;3 2;5 5;5 5];
>>val = [10.1 10.2 10.3 10.4];
>>A = accumarray(ind,val)
A =
```

```
    0   10.1000         0         0         0
    0         0         0         0         0
    0   10.2000         0         0         0
    0         0         0         0         0
    0         0         0         0   20.7000
```

矩阵 A 中的元素A(5,5)是由val(4)、val(5)相加而成。

若 ind = [1 2 3 8];val = [10.1 10.2 10.3 10.4];则在命令窗口输入如下程序:

```
>>ind = [1 2 3 8]';val = [10.1 10.2 10.3 10.4];
>>A = accumarray(ind,val)
A =

   10.1000
   10.2000
   10.3000
         0
         0
         0
         0
   10.4000
```

【例 1-5】 用 degree(度)表示角度向量 $\boldsymbol{x}$ = [15 30 45 60 75 90],列表显示三角函数 sin$\boldsymbol{x}$、cos$\boldsymbol{x}$ 和 tan$\boldsymbol{x}$。

解:

```
>>x = 15:15:90;                                %设置角度向量
>>table = [sind(x);cosd(x);tand(x)]'          %列表显示用角度的正弦、余弦和正切函数
table =

    0.2588    0.9659    0.2679
    0.5000    0.8660    0.5774
    0.7071    0.7071    1.0000
    0.8660    0.5000    1.7321
    0.9659    0.2588    3.7321
    1.0000         0       Inf
```

3. 程序编制的新特性

MATLAB7.0 在程序编制方面增加的新特性有 30 多项。例如用新的调用函数句柄的语法、嵌套函数、匿名函数、单元数组支持字符串函数、数据压缩支持 MAT 文件、为非双精度数据增加新的特性、新的数据输入/输出特性以及增加程序运算速度等。

下面着重介绍新的函数句柄的调用、匿名函数和嵌套函数。

【例 1-6】 已知抛物线方程 $y = ax^2 + bx + c$,求 $a = 1.3$, $b = 0.2$, $c = 30$, $x = 25$ 时的 y 值。

解:

在编辑器中写入

```
function y = para(a,b,c,x)                  %抛物线函数定义项
y = a* x.^2 + b* x + c;                     %函数体
```

在命令窗口调用抛物线函数

```
≫parah = @para;                             %调用抛物线函数句柄
≫parah (1.3, 0.2, 30, 25)                  %计算抛物线函数
ans =
847.5000
```

匿名函数（anonymous function）

匿名函数是提供一种快速建立简单函数的方法，而不用建立函数 M 文件，它在程序编制或在命令窗口直接调用时，带来很大的方便。匿名函数的编写格式如下：

fhandle = @(arglist) expr

式中，fhandle 为函数句柄；arglist 为变量列表，变量间用逗号分隔；expr 为有效的 MATLAB 数学表达式。

【例 1-7】 已知$\boldsymbol{x}=[1\ 2\ 3\ 4\ 5\ 6\ 7\ 8]$，求 $\boldsymbol{y}=\boldsymbol{x}^2$。

解：

在命令窗口键入如下程序：

```
≫sqr = @(x)x.^2;                           %创建匿名函数
≫x = 1:8;                                   %输入变量
≫y = sqr(x)                                 %计算变量 x 的二次方值
y =
1     4     9     16     25     36     49     64
```

【例 1-8】 计算数值积分 $q=\int_0^5 x^2\mathrm{d}x$。

解：

在命令窗口输入如下程序：

```
≫sqr = @(x)x.^2;                           %输入匿名函数
≫q = quad(sqr,0,5)                          %计算数值积分，积分限为[0,5]
q =                                         %计算数值积分结果
41.6667
```

【例 1-9】 用二重匿名函数，计算当 $c=2$ 或 $c=5$ 时的数值积分 $g=\int_0^1 (x^2+cx+1)\mathrm{d}x$。

解：

在命令窗口输入如下程序：

```
≫syms c                                     %设置符号变量 c
≫g = @(c)(quad(@(x)(x.^2 + c* x + 1),0,1));  %式中@(x)(x.^2 + c* x + 1)为匿名函
                                              数，@(c)…为二重匿名函数
```

```
g =
    @ (c) (quad (@ (x) (x.^2 + c* x + 1), 0, 1))
>>g (2)                                    %c = 2 时的数值积分
ans =
    2.3333
>>g (5)                                    %c = 5 时的数值积分
ans =
    3.8333
```

嵌套函数

嵌套函数是定义一个或几个函数包含在另外一个 MATLAB 函数内，这种内联函数称为嵌套函数。为了书写嵌套函数，简单的定义一个函数内的函数体包含另外的函数并用函数 M 文件编写。如同任何函数 M 文件一样，嵌套函数的组成部分的描述，应当符合函数 M 文件的基本要求。作为补充在嵌套函数的末尾必须添加 end 语句。嵌套函数的书写格式为

```
function x = A (p1, p2)
...
    function y = B (p3)
    ...
    end
...
end
```

【例 1-10】 绘制抛物线 $y = ax^2 + bx + c$，其中 $a = 1.3$，$b = 0.2$，$c = 30$，$x = [-25, 25]$。在这个例子中建立一个具有嵌套函数的函数句柄，然后将它传递到 MATLAB 的 fplot（绘制函数图形）函数，绘制抛物线。

解：

在编辑器窗口编制抛物线的嵌套函数如下：

```
function fhandle = makeparabola (a, b, c)
% MARKPARABOLA return a function handle with parabola coefficients
    fhandle = @parabola;            % @ is the function handle constructor
    function y = parabola (x)
    y = a* x.^2 + b* x + c;
    end
end
```

在命令窗口赋入数据和绘制函数图形如下：

```
>>h = makeparabola (1.3, 0.2, 30)          %赋予抛物线的系数 a、b、c 和函数句柄
h =
    @makeparabola/parabola
>>h (0)                                    %计算 x = 0 时的函数值
ans =
    30
```

```
≫h (25)                                    %计算 x = 25 时的函数值
ans =
    847.5000
≫fplot (h,[-25,25])                        % 绘制抛物线图形，取值
                                             为[-25,25]，如图 1-20
                                             所示
≫grid on                                   %添加坐标栅格线
```

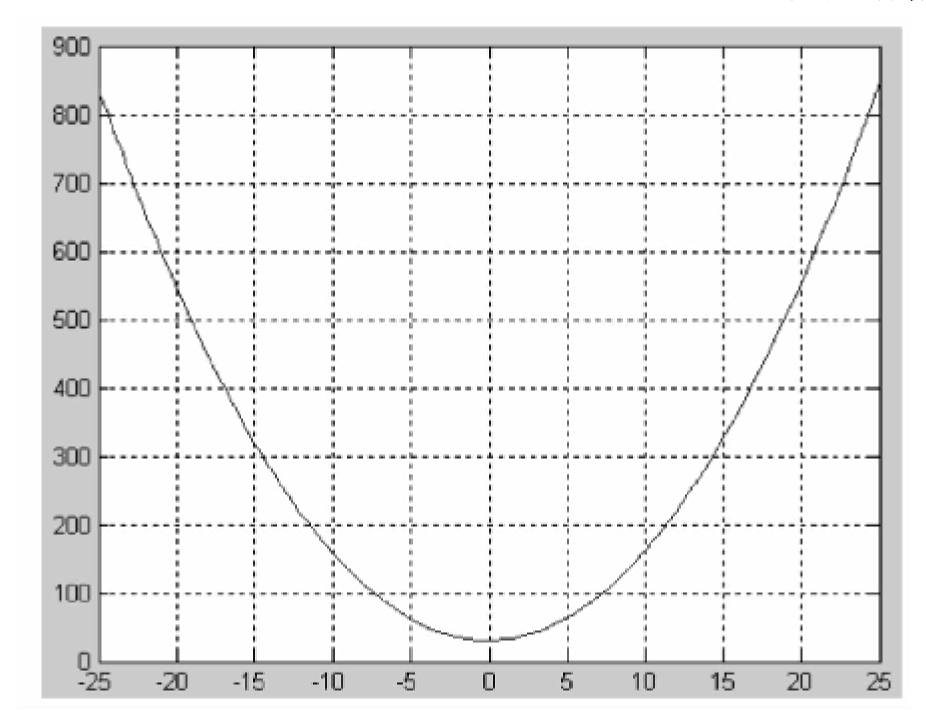


图 1-20　抛物线图形

4. 图形和三维可视化新的特性

MATLAB7.0 对图形的绘制和编辑有 12 项改进。如绘图工具、图形代码的产生、数据的检查工具、图形的注解特性、新的坐标轴特性和新的根对象特性等。本节介绍比较有用的绘图工具和数据检查工具。

(1) 绘图工具

MATLAB 提供了一个集成的绘图工具，它具有人机交互式的绘图环境。这个环境下能建立多种类型的图形。当选择好在工作空间浏览器中变量，即可直接绘图。你可以在图形中添加注解，如剪头、辅助线、图例、坐标和说明文本等。可以容易地建立和操作子图以及设置曲线图特性，如线宽、标记点大小和线的颜色等。

为了建立图形工具，可以使用 plottools 命令，启动图形编辑窗口，也可以从工作空间浏览器中，选择变量，点击 plot tools 图标。

在命令窗口输入如下程序，则可得图 1-21 所示的正弦曲线。

```
≫x = 0: pi/9: pi;                                          %设置角度向量
≫y = sin(x);                                               %计算正弦函数
≫plot (y,'DisplayName','y','YDataSource','y'); figure(gcf)  %绘制正弦波曲线
≫grid on                                                   %增加坐标格栅线
```

```
≫plottools                                        %启动绘图工具
```

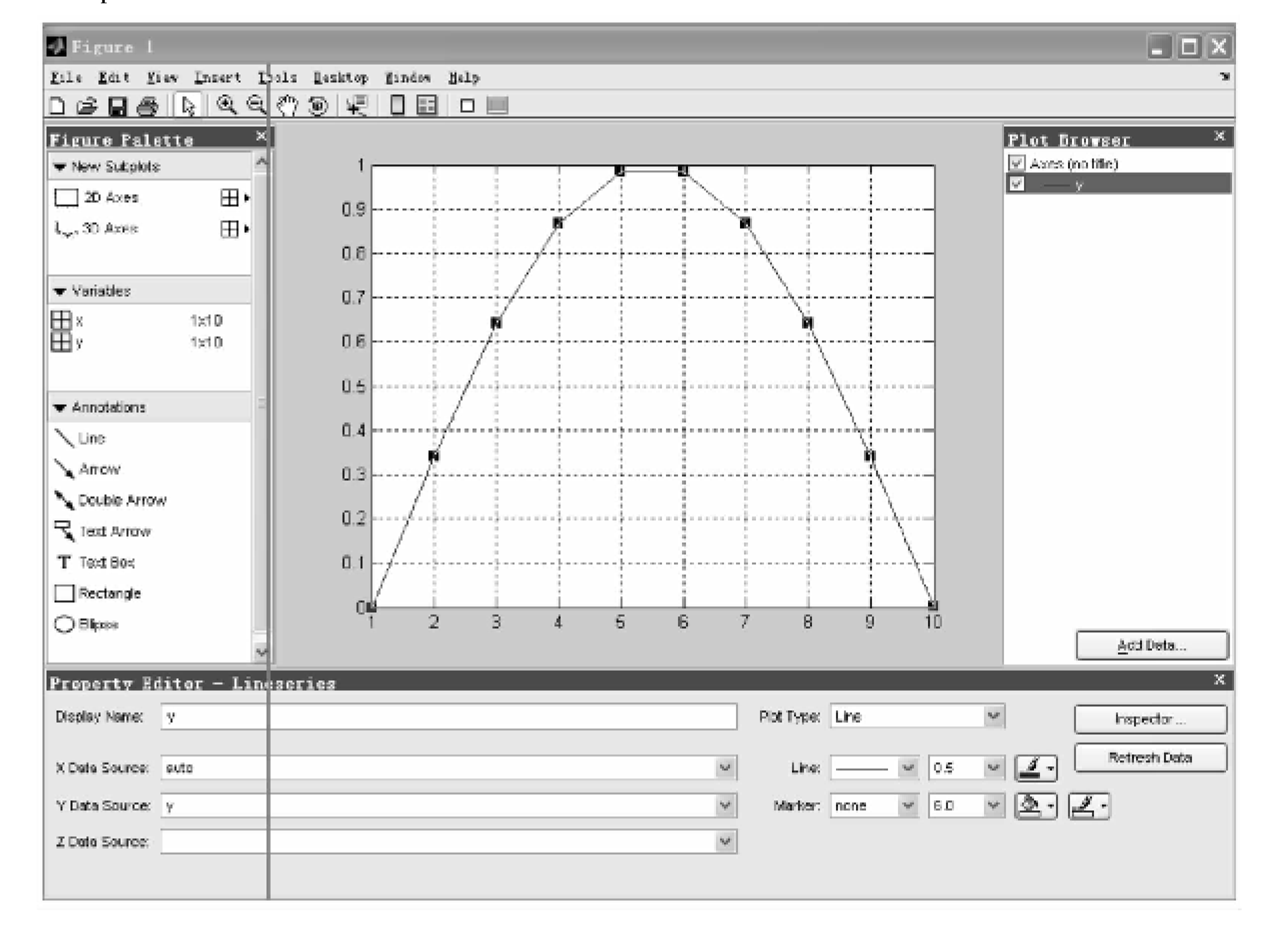


图 1-21　正弦曲线图及图形编辑工具

在图 1-21 中左侧为图形调色板，中间为图形显示器，右侧为绘图浏览器，下方为特性编辑器。

(2) 数据检测工具

MATLAB7.0 中的检测工具有：

数据光标——当点击图形编辑工具中的图标 (data cursor) 即产生十字光标，将十字光标移到图形曲线的相应点上，并点击，即自动显示该点的数据。

图形的缩放——可以使用图形编辑工具中的 Zooming 图标，来缩放二维和三维的图形。

图形的平移——可以使用图形编辑工具中的 Panning 图标，实现图形的平移。

三维立体图的旋转——可以使用权图形编辑工具中 Rotate 3D，实现三维视图的旋转。

照相机工具条——可以通过鼠标操纵三维立体图的视角。

5. 外部接口/应用程序接口

MATLAB 能够实现与下列计算机语言的综合应用与编程：

C/C++	Java
Visual Basic	Excel
COM	Web

6. 图形化用户接口

增加了新的控制元件和工具，改进了菜单编辑器。

7. 增加了新的工具箱

与 MATLAB6.5 版本相比较，增加了生物仪器工具箱、数据取得工具箱、模型预校正工具箱、OPC 工具箱和 RF 工具箱等。

1.6 MATLAB 的符号

MATLAB 有算术运算符、关系运算符、逻辑运算符、特殊运算符、位操作运算符和设置运算符等共 61 种，为便于查找，把各符号的含义列表介绍如下。

1.6.1 算术运算符

算术运算符用于数组或矩阵的四则运算，见表 1-3。

表 1-3 算术运算符

名　称	说　明	符　号
plus	加法	+
uplus	非数组加法	+
minus	减法	-
uminus	非数组减法	-
mtimes	矩阵乘法	*
times	数组乘法	.*
mpower	矩阵乘幂	^
power	数组乘幂	.^
mldivide	矩阵左除	\
mrdivide	矩阵右除	/
ldivide	数组左除	.\
rdivide	数组右除	./
kron	Kronecker 向量积	kron

1.6.2 关系运算符

用于程序中变量间关系的判断，或数学表达式中的关系表示，见表 1-4。

表 1-4 关系运算符

名　称	说　明	符　号
eq	相等	==
ne	不等	~=
lt	小于	<
gt	大于	>
le	小于等于	<=
ge	大于等于	>=

1.6.3 逻辑运算符

用于程序中变量间逻辑关系的判断，见表 1-5。

表 1-5　逻辑运算符

名　　称	说　　明	符　　号
and	逻辑与	&
or	逻辑或	|
not	逻辑非	~
xor	逻辑异或	
any	假如任何向量元素非零则为真	
all	假如所有向量元素非零则为真	

1.6.4 特殊运算符（见表 1-6）

表 1-6　特殊运算符

名　　称	说　　明	符　　号
colon	冒号	:
paren	括号和下标	( )
paren	方括号	[]
paren	大括号和下标	{ }
punct	建立句柄	@
punct	小数点	.
punct	访问结构场	.
punct	父目录	..
punct	连续写入	...
punct	逗号	,
punct	分号	;
punct	注解	%
punct	调用操作系统命令	!
punct	赋值	=
punct	引号	‘
transpose	转置	.’
ctranspose	共轭复数转置	‘
horzcat	水平连接	[,]
vertcat	垂直连接	[;]
subsasgn	下标配置	( ), { }
subsref	下标参数	( ), { }
subsindex	下标索引	

1.6.5 位操作符（见表 1-7）

表 1-7 位操作符

名　称	说　明
bitand	按位相与
bitcmp	补码
bitor	按位相或
bitmax	最大浮点整数
bitxor	按位异或
bitset	设置某位
bitget	提取某位
bitshift	按位移位

1.6.6 设定操作符（见表 1-8）

表 1-8 设定操作符

名　称	说　明
union	设定合集
unique	设定惟一
intersect	设定交集
setdiff	设定差集
setxor	设定异或
ismember	检测向量中是否包含某元素，或矩阵中是否有相同的行

以上符号都可以通过 help 命令来取得对操作符的解释。例如：

```
≫help ismember
ISMEMBER True for set member.
    ISMEMBER (A,S) when A is a vector returns a vector containing 1 where
    the elements of A are in the set S and 0 otherwise. A and S can be
    cell arrays of strings.
    ISMEMBER (A,S,'rows') when A and S are matrices with the same
    number of columns returns a vector containing 1 where the rows of
    A are also rows of S and 0 otherwise.
    See also UNIQUE, INTERSECT, SETDIFF, SETXOR, UNION.
Overloaded methods
    help cell/ismember.m
```

根据上述说明，ismember 是用于判别向量或矩阵是否含有设定元素，对于形式为 ismember (A,S)，当 A 是向量，S 为元素，则返回一个同维向量。它包含 1 时，则表示 A 中的对应元素含有 S，否则置零则表示不含有 S。

ismember (A,S,'rows')，则用于判别维数相同的矩阵 A、S 中是否含有相同的行。它返

回一个列向量，若 A 和 S 对应的行相同，则该列对应元素置 1，否则置 0。

现用 ismember 编程举例如下：

```
≫A = [13 8 5 7 24 11 7 55];                 %设已知向量
≫ismember (A,7)                              %向量 A 中是否含有 7 这个元素
ans =                                        %答：A 中第 4、7 列，有 7 这个元素。ans
                                               是当表达式答案未给命名时，自动建立的
                                               变量名，故可认为是默认的变量。在本式
                                               中代表 ismember (A,7)
     0     0     0     1     0     0     1     0
≫A = [1 3 5 7 9;2 3 5 8 13;3 4 7 11 18]      %已知 3 行 5 列的矩阵 A
A =
         1     3     5     7     9
         2     3     5     8    13
         3     4     7    11    18
≫S = [1 3 5 7 9;2 3 5 8 13;4 7 11 25 3 ]     %已知 3 行 5 列矩阵 S
S =
         1     3     5     7     9
         2     3     5     8    13
         4     7    11    25     3
≫ismember(A,S,'rows')                        %A 和 S 中是否有相同的行
ans =                                        %答：A 和 S 中有第 1、2 行相同
     1
     1
     0
```

1.7 MATLAB 中的常用命令

MATLAB 常用命令的名称和功能见表 1-9。

表 1-9 MATLAB 常用命令的名称和功能

命令名称	功能及说明
cd	显示或改变当前工作目录
clc	清命令窗口，并使光标返回
clear	清工作空间的变量
clf	清当前图形
delete	删除文件或图形对象
dir	列出当前目录下文件名
disp	显示数组或字符串变量
doc	在帮助浏览器中寻找相关的 html 文件

（续）

命令名称	功能及说明
diary	保存 MATLAB 运行期间的文本
help	在线帮助，对 MATLAB 的所有函数、命令、符号的功能和用法提供帮助
lookfor	搜索所有 M 文件，寻找相关的关键词
load	从磁盘加载指定变量文件到工作空间
path	显示目录路径
save	保存工作空间变量到磁盘，即建立后缀为 MAT 的数据文件
type	列出 M 文件的内容
what	列出在指定目录中的 MATLAB 文件
who	列出在当前工作空间的变量
whos	详细列出在当前工作空间的变量大小、位数和数据类型
which	显示函数或文件所处的路径名
Ctrl home	光标移至页首
Ctrl e	光标移至行末
End	光标移至行末

在 MATLAB 中，最有用的命令之一是 help，它能提供所有函数、命令、符号的功能和用法，并举例给予说明。在其他计算机软件中很少有 MATLAB 那样详尽的帮助功能。

【例 1-11】 寻求对函数 diag 的帮助。

解：

在命令窗口输入如下程序：

```
>>help diag
DIAG Diagonal matrices and diagonals of a matrix.
    DIAG (V,K) when V is a vector with N components is a square matrix
    of order N + ABS (K) with the elements of V on the K-th diagonal. K = 0
    is the main diagonal, K > 0 is above the main diagonal and K < 0
    is below the main diagonal.
    DIAG (V) is the same as DIAG (V,0) and puts V on the main diagonal.
    DIAG (X, K) when X is a matrix is a column vector formed from
    the elements of the K-th diagonal of X.
    DIAG (X) is the main diagonal of X. DIAG (DIAG (X)) is a diagonal matrix.
    Example
        m = 5;
        diag (-m:m) + diag (ones (2*m,1),1) + diag (ones (2*m,1),-1)
    produces a tridiagonal matrix of order 2*m + 1.
    See also spdiags, triu, tril.
    Overloaded functions or methods (ones with the same name in other directories)
        help gf/diag.m
        help sym/diag.m
```

Reference page in Help browser

doc diag

在这段帮助中，说明 diag 是用于求对角线矩阵或矩阵的对角线。当 V 是向量，含有 n 个元素，那么 diag（V，K）是一个阶次为 n + abs（K）的方阵，而向量 V 的元素，是处于矩阵的第 K 根对角线上。当 K = 0，则 V 的元素在主对角线上，当 K > 0，则 V 的元素在主对角线以上。当 K < 0，则 V 的元素在主对角线以下。

diag（V）与 diag（V，0）是相同的。

diag（X，K），当 X 是矩阵，则它是 X 矩阵的第 K 根对角线组成的列向量。diag（X）是矩阵 X 的主对角线所组成的列向量，而 diag（diag（X））则是矩阵 X 的一根主对角线组成的矩阵。

举例 m = 5，有

diag（ – m：m） + diag（ones（2 * m，1），1） + diag（ones（2 * m，1），– 1）

产生阶次为 2m + 1 的 3 根对角矩阵。

参看 spdiags（疏稀对角矩阵）、triu（提取矩阵的上三角）、tril（提取矩阵的下三角）额外的功能或使用方法参见

help gf/diag.m

help sym/diag.m

参考浏览文件为

doc diag

另一个重要的命令是 type，它能列出 M 文件（有关 M 文件请见第 3 章）的内容。通过文件的内容可以了解 M 文件的构成、使用的条件和学习编程的技巧。

【例 1-12】 列出平均值计算函数 mean 的 M 文件。

解：

在命令窗口键入下列命令：

```
>>type mean
    function y = mean (x,dim)
%MEAN   Average or mean value.
%   For vectors, MEAN (X)is the mean value of the elements in X. For
%   matrices, MEAN (X)is a row vector containing the mean value of
%   each column.For N-D arrays, MEAN (X)is the mean value of the
%   elements along the first non-singleton dimension of X.
%
%   MEAN (X,DIM)takes the mean along the dimension DIM of X.
%
%   Example: If X = [0 1 2;3 4 5]
%
%   then mean (X,1) is [1.5 2.5 3.5] and mean(X,2) is [1;4]
%
%   See also MEDIAN,STD,MIN,MAX,COV.
```

```
%   Copyright 1984 - 2001 The MathWorks, Inc.
%   $Revision: 5.16$ $Date: 2001/04/15 12: 01: 26 $

if nargin == 1,
   % Determine which dimension SUM will use
  dim = min(find(size(x) ~ = 1));
   if isempty(dim), dim = 1; end
   y = sum(x)/size(x,dim);
else
  y = sum(x,dim)/size(x,dim);
end
```

在这个文件中，包含了函数定义行、帮助文本的内容、函数体和附注等。

另一个重要的命令是 lookfor，它能在不确切知道函数 M 文件的拼写时，搜索相关的关键词，从中寻找所需确切的函数。

【例 1-13】 用 lookfor substitution 来搜索变量替换的函数。

解：

在命令窗口键入 lookfor substitution，则显示如下内容：

```
≫lookfor substitution
SUBS   Symbolic substitution.
SUBS   Symbolic substitution. Also used to evaluate f numerically.
```

由此说明 subs 可以用于变量替换。但由于 lookfor 命令需要在大量的 M 文件中搜索帮助文本中的词汇与搜索词相匹配，所以搜索时间较长。

另一个具有帮助功能的命令是 doc 函数名。它是从帮助浏览器中寻找有关函数名的 html（超文本链接标记语言）文件。它是属于参考文件，在该文件中，它对函数的语法、函数的定义和应用举例都作了说明。下面举例予以说明。

【例 1-14】 用 doc std 来寻找标准偏差的参考文件。

解：

在命令窗口键入 doc std，则显示如下文本（见图 1-22、图 1-23）：

在这里面，该文件对 std 的名称、语法、定义、说明、举例和相关函数提供了帮助。对于不确切知道函数名的拼法时，还可以模糊函数名加双击 Tab 键来寻找。

【例 1-15】 求与多项式有关的函数，模糊名为 pol。

解：

在命令窗口键入 pol，随后再双击 Tab 键，对于 MATLAB6.x 版本，则显示如下：

```
≫pol
pol2cart      poly2rc      polybool      polyfit      polystab
polar         poly2str     polycon       polyform     polytool
polcmap       poly2sym     polyconf      polyint      polyval
polsim        poly2tfd     polycut       polyjoin     polyvalm
```

MATLAB Function Reference

std

Standard deviation

Syntax

```
s = std(X)
s = std(X,flag)
s = std(X,flag,dim)
```

Definition

There are two common textbook definitions for the standard deviation s of a data vector X.

$$(1)\quad s = \left(\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{\frac{1}{2}}$$

$$(2)\quad s = \left(\frac{1}{n}\sum_{i=1}^{n}(x_i-\bar{x})^2\right)^{\frac{1}{2}}$$

where

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n}x_i$$

and n is the number of elements in the sample. The two forms of the equation differ only in $n-1$ versus n in the divisor.

图 1-22　标准偏差 std 的参考文本之一

Description

s = std(X), where X is a vector, returns the standard deviation using (1) above. If X is a random sample of data from a normal distribution, s^2 is the best *unbiased* estimate of its variance.

If X is a matrix, std(X) returns a row vector containing the standard deviation of the elements of each column of X. If X is a multidimensional array, std(X) is the standard deviation of th elements along the first nonsingleton dimension of X.

s = std(X,flag) for flag = 0, is the same as std(X). For flag = 1, std(X,1) returns the standard deviation using (2) above, producing the second moment of the sample about its mean.

s = std(X,flag,dim) computes the standard deviations along the dimension of X specified by scalar dim.

Examples

For matrix X

```
X =
      1     5     9
      7    15    22
s = std(X,0,1)
s =
    4.2426    7.0711    9.1924
s = std(X,0,2)
s =
    4.000
    7.5056
```

See Also

corrcoef, cov, mean, median

startup　　　　stem

图 1-23　标准偏差 std 的参考文本之二

poly　　poly2th　　polydec　　polymerge　　polyxpoly
poly2ac　　poly2trellis　　polyder　　polyscale

poly2lsf　　polyarea　　polyeig　　polysplit

对于 MATLAB7.0 版本，则在单击 Tab 键后显示带滑动杆的有 pol 开头的函数索引表。由此显示出与 pol 相近的函数有 33 个，用户可在供选取的 33 函数中寻找确切的与多项式有关的函数，或用 help 命令查询某函数的确切用法。

【例 1-16】 查询模糊名为 is，其部分含义为“是属于 xxxxx 吗?”

解:

对于 MATLAB6.x 版本，在命令窗口键入 is，随后再双击 Tab 键，则显示如下：

```
>>is
is2rc
is95cohdetector
is95framequalitydet
is95fwdchcodec
is95fwdchcodec2
is95fwdchdescramble
is95fwdchdetection
is95fwdchdetection2
is95fwdchendtoend
is95fwdchendtoend2
is95fwdchinterleaver
is95fwdchpwrbitext
is95fwdchrakefinger
is95fwdchrepeater
is95longcodegenerator
is95noncohdetector
is95revchcodec
is95revchcodec2
is95revchdetection
is95revchdetection2
is95revchinterleaver
is95revchmodspread
is95revchrepeater
is95revchtransmitter
is95revchtransmitter2
isa
isafin
isappdata
isbusday
isbw
iscatter
isind
isinf
isjava
iskeyword
isletter
islogical
islsys
ismap
ismapped
ismembc
ismember
isnan
isnode
isnumeric
iso2geod
isobject
isocaps
isocolors
isonormals
isosurface
ispc
isppc
ispref
isprime
ispsys
isreal
isrgb
isruntime
issame
isslfcjg
issourcecontrolconfigured
```

iscell
iscellstr
ischannel
ischar
iscvar
isdioline
isdir
isempty
isequal
isfield
isfinite
isfis
isglobal
isgray
ishandle
ishold
isieee
isspace
issparse
isstr
isstruct
isstudent
issystem
isthss
istnode
istree
istrellis
isunix
isvarname
isvms
iswt
iswt2
isymm

对于 MATLAB7.0 版本，则在单击 Tab 键后显示带滑块的、有 is 开头的函数索引表。以上共有 95 个与 is 有关的函数供选用。对于其中常用函数见表 1-10。

表 1-10　有关 is 的常用函数

序　号	函 数 名	说　明
1	isa	假若对象是指定等级，则为真
2	isbusday	假若是交易日，则为真
3	isempty	假若是空矩阵，则为真
4	isequal	假若数组是数值相等，则为真
5	isglobal	假若是全局变量，则为真
6	ishold	假若状态是保持，则为真
7	isjava	假若是 java 数组，则为真
8	isletter	假若是字母，则为真
9	islogical	假若是逻辑符，则为真
10	ismember	假若某元素属于指定向量，则为真
11	isprime	假若是质数，则为真
12	isreal	假若是实数向量，则为真
13	isstr	假若是字符串，则为真
14	isstruct	假若是结构数组，则为真
15	isvarname	假若是有效的变量名，则为真

由于 MATLAB 的强大功能，在各技术领域的应用广泛，因此从事科学计算和技术设计工作的读者对 MATLAB 十分感兴趣。目前有关 MATLAB 的书籍不下数十种，作者们都在不同侧面介绍 MATLAB 及其应用。学习的目的是为了应用，但是有的读者认为 MATLAB

的软件中有1000多条数学和工程函数，学习中觉得比较枯燥和繁琐，在解决具体问题时仍难以下手。这主要是对实际问题的分析还不够，对MATLAB的指令、语句和函数的熟练程度还不够，对编程技巧尚缺乏所造成的。如果能应用MATLAB解决一两个实际问题，那么会发现MATLAB是个非常有用的工具，分析计算和图形绘制是一种轻松愉快的事，并且促使进一步掌握它。

本书是一本关于MATLAB的入门书又是一本应用书，它既介绍基础知识，又介绍它的应用。它是使用MATLAB的命令、语句和函数以及通过编制程序，叙述解题过程和方法。作者试图通过大量例题使读者迅速掌握MATLAB的各种命令、语句和函数的用法。从例子中领会命令、语句和函数的含义和用法。读者可以通过上机操作，逐步熟悉MATLAB的命令、语句和函数的用法。如果对某个命令和函数不理解，那么可以用help命令、doc命令或type命令去查询该命令或函数的用法以及查看程序的内容，或者通过lookfor命令去查找相关命令和函数。通过使用MATLAB，它帮助你在工作中提高效率，从繁琐的报表数字和图形绘制的手工劳动中解放出来，使数学运算和图形绘制走向自动化。

由于MATLAB的基础是矩阵，该软件要求读者熟练掌握矩阵的运算。所以对不熟悉线性代数的读者，建议复习线性代数是有益的。

第 2 章　MATLAB 与线性代数

MATLAB 的基础是线性代数，而 MATLAB 软件建立后又促进了线性代数的应用和发展。本章介绍数组和矩阵的列写方法，矩阵的运算、矩阵的转置、矩阵的行列式、矩阵的乘幂、矩阵的秩、矩阵的范数、矩阵的条件数、逆矩阵计算、广义逆矩阵的计算、矩阵的分解、矩阵的特征值、特征向量、单位矩阵、矩阵的大小、Pascal 矩阵、魔方矩阵、随机矩阵、Hilbert 矩阵、稀疏矩阵等。以下各节列出了 MATLAB 有关线性代数的内容，标有“%”的为注解，供阅读参考，上机操作时可不用输入。

2.1　数组的表示，冒号的用法

数组是由若干个数排列成矩形的组合。数组的内涵经过扩充，范围很广。以维数来分，有一维数组、二维数组和多维数组。按数组中元素来分，有数值数组、字符串数组、结构数组和单元数组等。

数组的表示往往与冒号联合在一起。冒号用于表示向量、带有下标的数组以及用来表示循环。

j:k　　　相当于向量[j,j+1,j+2,…,k]

j:i:k　　相当于向量[j,j+i,j+2*i,…k]

A(:,j)　　矩阵 A 的 j 列

A(i,:)　　矩阵 A 的 i 行

这里 i、j、k 必须为标量。

下面是最简单的一维数值数组的例子。

【例 2-1】　建立从 1～100、步长为 3 的数组。

解:

在命令窗口键入如下程序:

```
>>x = 1:3:100          %一维数值数组表示法，初值：步长：终值，中间用冒号分开。“>>”为
                         MATLAB 在命令窗口的提示符，x 为设置的数组名，键入后回车
x =                    %命令窗口的输出响应
  Columns 1 through 13
  1    4    7    10    13    16    19    22    25    28    31    34    37
  Columns 14 through 26
  40   43   46   49    52    55    58    61    64    67    70    73    76
  Columns 27 through 34
  79   82   85   88    91    94    97    100
```

【例 2-2】　建立矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3 & 4 & 5\\ 6 & 7 & 8 & 9 & 10\\ 11 & 12 & 13 & 14 & 15\end{bmatrix}$。

解:

```
>>A(1,:)=1:5                    %设置矩阵的第一行
A =
     1     2     3     4     5
>>A(2,:)=6:10                   %设置矩阵的第二行
A =
     1     2     3     4     5
     6     7     8     9    10
>>A(3,:)=11:15                  %设置矩阵的第三行，设置完成
A =
     1     2     3     4     5
     6     7     8     9    10
    11    12    13    14    15
```

【例 2-3】 计算数组 1，2，3，…，99，100 之和。

解:

```
>>s=0;                          %设置初值
>>for i=1:100                   %建立 100 次循环
      s=s+i;                    %累加
  end,s                         %结束循环，并显示累加结果
s =
        5050
```

2.2 线性间隔向量

产生 1 个行向量，从 x_1 到 x_2 之间，均匀分布 n 个数。书写格式为 linspace(x1,x2,n)。

【例 2-4】 用 linspace 列出 1 ~ 5 之间的 20 个等距数组。

解:

在命令窗口键入如下程序:

```
>>x=linspace(1,5,20)          %以初值、终值、组数来表示，x 为数组名
x =                           %命令窗口的输出响应
Columns 1 through 7
1.0000    1.2105    1.4211    1.6316    1.8421    2.0526    2.2632
Columns 8 through 14
2.4737    2.6842    2.8947    3.1053    3.3158    3.5263    3.7368
Columns 15 through 20
3.9474    4.1579    4.3684    4.5789    4.7895    5.0000
```

2.3 对数化间隔向量

产生 1 个行向量，从 10^{d_1} 到 10^{d_2} 之间以对数刻度分布的 n 个数。书写格式为 logspace(d1,d2,n)，这里 d1、d2 和 n 必须是标量。

【例 2-5】 用 logspace 产生 10 ~ 100 间以对数刻度分布的 12 个数。

解：

在命令窗口键入如下程序：

```
≫x = logspace(1,2,12)    %1、2 为 10 的乘幂，作为数组的始点和终点，12 为组数，x 为数
                          组名
x =
Columns 1 through 11
10.0000    12.3285    15.1991    18.7382    23.1013    28.4804    35.1119
43.2876    53.3670    65.7933    81.1131
Column 12
100.0000
```

2.4 显示格式的设置

可以用表 2-1 所示的 MATLAB 命令来设置数字显示格式。

表 2-1 显示格式的设置命令

MATLAB 命令	说　明
FORMAT	默认格式，与 5 位定点计数制相同
FORMAT SHORT	5 位定点计数制
FORMAT LONG	15 位定点计数制
FORMAT SHORT E	5 位浮点计数制
FORMAT LONG E	15 位浮点计数制
FORMAT SHORT G	最佳定点或 5 位浮点
FORMAT LONG G	最佳定点或 15 位浮点
FORMAT HEX	十六进制格式
FORMAT +	只显示 +、－和空格符号，而对正、负和零元素的虚部是忽略
FORMAT BANK	银行格式，以元、分定点
FORMAT RAT	近似以分式表示
FORMAT COMPACT	紧凑格式，段落之间不设置空行
FORMAT LOOSE	松散格式，为段前设空行

【例 2-6】 用 format short，format long，format rat 分别显示 π 的值。

解：

```
≫format short           %设置定点格式，显示 5 位
≫pi                     %显示 π 的值
  ans =                 %ans 是自动建立的变量名，亦即默认的变量名。在本例中，
                          表示常量 π
  3.1416
≫format long            %设置定点格式，显示 15 位
≫pi                     %显示 π 的值
```

```
    ans =
    3.14159265358979
>>format rat                           %设置近似分式显示格式
>>pi                                   %显示π的值
    ans =
    355/113
```

2.5 矩阵的加法与减法

矩阵是由若干个行与列的数字作矩形排列。矩阵的行数和列数相同的两个矩阵的加法和减法，返回1个同样行数和列数的矩阵，其中每个元素为原先两个矩阵对应元素之和，或两个矩阵对应元素之差。若行数和列数不同的两个矩阵的加法和减法，则显示出错。

【例2-7】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 10\end{bmatrix}$ 与 $\boldsymbol{B}=\begin{bmatrix}1 & 3 & 5\\7 & 9 & 11\\13 & 15 & 16\end{bmatrix}$，求矩阵 $\boldsymbol{B}-\boldsymbol{A}$，$\boldsymbol{A}+\boldsymbol{B}$。

解：

在命令窗口键入如下程序：

```
>>A = [1 2 3;4 5 6;7 8 10]      %设A为已知矩阵。矩阵的书写方法是：数与数之间用逗号
                                 或空格分开，换行时用分号分开，矩阵的开始和终了用方
                                 括号
A =
      1          2          3
      4          5          6
      7          8         10
>>B = [1 3 5;7 9 11;13 15 16]   %设已知矩阵B
B =
      1          3          5
      7          9         11
     13         15         16
>> B - A                         %阶数相同的矩阵B与A之差。矩阵之差的矩阵元素是B和
                                 A对应元素之差
ans =
      0      1      2
      3      4      5
      6      7      6
>>A + B                          %同阶矩阵B与A之和。矩阵之和的矩阵元素是B和A对
                                 应元素之和
ans =
          2          5          8
```

```
    11        14        17
    20        23        26
```

2.6 数组的乘法与除法

数组的乘法和除法，返回为 1 个同维的数组，其数组的元素为两组数组间对应元素的相乘和相除。若两数组的维数各不相同，则进行数组乘、除法时，显示出错。

【例 2-8】 已知 3 阶方阵 $\boldsymbol{A}=\begin{bmatrix}1&2&3\\2&4&6\\3&6&12\end{bmatrix}$，$\boldsymbol{B}=\begin{bmatrix}1&2&3\\1&2&3\\1&2&3\end{bmatrix}$，求方阵 $\boldsymbol{A}$、$\boldsymbol{B}$ 的数组乘法和除法。

解：

在命令窗口键入如下程序：

```
>>A = [1 2 3;2 4 6;3 6 12]          %设 A 为已知矩阵
A =

    1         2         3
    2         4         6
    3         6        12

>>B = [1 2 3]                       %设 B 为 1 维向量
B =
    1      2      3
>>B = [B;B;B]                       %设已知矩阵 B，山 3 行上述 B 向量组成，此时 B 向量已被 B
                                      矩阵覆盖
B =

    1         2         3
    1         2         3
    1         2         3

>>A. * B                            %矩阵 A、B 的数组积为矩阵对应元素的相乘，它与矩阵乘法
                                      是不同的，注意在数组乘法的乘号 * 前加了句号 “.”
ans =                               %A、B 两数组之积，注意这与下面的矩阵乘法是不同的

    1         4         9
    2         8        18
    3        12        36

>>A * B                             %矩阵 A、B 的积
ans =

    6        12        18
   12        24        36
   21        42        63
```

```
>>A./B                                        %矩阵 A 被矩阵 B 除的数组除法，即对应元素相除，注意斜杠前的句号。这与下面的矩阵除法是不同的
ans =
     1     1     1
     2     2     2
     3     3     4
>>A/B                                         % 矩阵 A 右除矩阵 B
Warning: Matrix is singular to working precision.   % 警告：矩阵在工作精确度内是奇异的，因为矩阵 A、B 均为奇异矩阵，所谓奇异矩阵为矩阵的行列式接近为零
ans =                                         %答：矩阵每个元素均趋向无限大(inf)
   Inf   Inf   Inf
   Inf   Inf   Inf
   Inf   Inf   Inf
```

2.7 矩阵的乘法

设矩阵 $\boldsymbol{A}$ 是一个 $m \times p$ 的矩阵（m 行 p 列），矩阵 $\boldsymbol{B}$ 是一个 $p \times m$ 的矩阵（p 行 n 列），即

$$\boldsymbol{A} = (a_{i,j})_{m \times p} \qquad \boldsymbol{B} = (b_{i,j})_{p \times m}$$

则矩阵的乘积（m 行 n 列）

$$\boldsymbol{C} = \boldsymbol{AB} = (c_{i,j})_{m \times n} \tag{2-1}$$

$$c_{i,j} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^{p} a_{i,k}b_{k,j} \tag{2-2}$$

式中，$i = 1, 2, \cdots, m$；$j = 1, 2, \cdots, n$；$k = 1, 2, \cdots, p$。

必须注意，矩阵相乘，矩阵 $\boldsymbol{A}$ 的列数应等于矩阵 $\boldsymbol{B}$ 的行数，否则将显示出错。

【例 2-9】 已知 3 阶矩阵 $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{bmatrix}$ 和 $\boldsymbol{B} = \begin{bmatrix} 1 & 3 & 5 \\ 7 & 9 & 11 \\ 13 & 15 & 17 \end{bmatrix}$，求矩阵的积。

解：

```
>>A = [1 2 3;4 5 6;7 8 10]                    %输入矩阵 A
A =
     1     2     3
     4     5     6
     7     8    10
>>B = [1 3 5;7 9 11;13 15 17]                 %输入矩阵 B
B =
```

```
     1         3         5
     7         9        11
    13        15        17
```

```
>>C = A* B                                  %矩阵 A、B 的积为 C
C =

     54        66        78
    117       147       177
    193       243       293
```

2.8 矩阵的左除

矩阵的左除为矩阵乘法的逆运算，若 $\boldsymbol{AB} = \boldsymbol{C}$ 则 $\boldsymbol{B} = \boldsymbol{A} \backslash \boldsymbol{C}$，即 $\boldsymbol{B}$ 等于 $\boldsymbol{A}$ 左除 $\boldsymbol{C}$。注意矩阵左除用反斜杠表示。矩阵的左除，常用于解线性方程组，$\boldsymbol{AX} = \boldsymbol{B}$。这时 $\boldsymbol{X}$ 的解就等于矩阵 $\boldsymbol{A}$ 左除矩阵 $\boldsymbol{B}$。

【例 2-10】 已知矩阵 $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 10 \end{bmatrix}$ 和 $\boldsymbol{C} = \begin{bmatrix} 54 & 66 & 75 \\ 117 & 147 & 171 \\ 193 & 243 & 283 \end{bmatrix}$，求矩阵 $\boldsymbol{A}$ 左除 $\boldsymbol{C}$，即 $\boldsymbol{A} \backslash \boldsymbol{C}$。

解：

```
>>A = [1 2 3;4 5 6;7 8 10]                  %输入矩阵 A
A =

     1         2         3
     4         5         6
     7         8        10

>>C = [54 66 75;117 147 171;193 243 283]    %输入矩阵 C
C =

     54        66        75
    117       147       171
    193       243       283

>>A \C                                      %矩阵 C 被矩阵 A 左除，即为 A* ans = C
                                              的逆运算

ans =

     1         3         5
     7         9        11
    13        15        16
```

2.9 矩阵的右除

矩阵的右除也为矩阵乘法的逆运算，但所求解矩阵的位置不同，若 $\boldsymbol{AB} = \boldsymbol{C}$，则 $\boldsymbol{A} = \boldsymbol{C}/\boldsymbol{B}$，即矩阵 $\boldsymbol{A}$ 等于 $\boldsymbol{C}$ 右除 $\boldsymbol{B}$，注意右除用斜杠。

【例 2-11】 已知矩阵 $\boldsymbol{B}=\begin{bmatrix}1 & 3 & 5\\7 & 9 & 11\\13 & 15 & 16\end{bmatrix}$，$\boldsymbol{C}=\begin{bmatrix}54 & 66 & 75\\117 & 147 & 171\\193 & 243 & 283\end{bmatrix}$，求矩阵 $\boldsymbol{C}$ 被矩阵 $\boldsymbol{B}$ 右除，即 $\boldsymbol{C/B}$。

解：

```
C = [54 66 75;117 147 171;193 243 283]   %输入矩阵 C
C =

    54    66    75
   117   147   171
   193   243   283

>>B = [1 3 5;7 9 11;13 15 16]            %输入矩阵 B
B =

     1     3     5
     7     9    11
    13    15    16

>>C/B                                    %矩阵 C 右除矩阵 B，即为 ans* B = C 的逆运算
ans =

     1     2     3
     4     5     6
     7     8    10
```

2.10 方阵的行列式

当矩阵的行数和列数相等时，称为方阵。方阵的行列式用 det$\boldsymbol{x}$ 表示，其中 $\boldsymbol{x}$ 为方阵。

$$\det x = \sum (-1)^{r(j_1, j_2, \cdots, j_n)} a_{1j_1} a_{2j_2} \cdots a_{nj_n}$$

式中，$r(j_1, j_2, \cdots, j_n)$为逆序数，$\sum$表示对所有 n 阶排列$j_1, j_2, \cdots, j_n$ 求和，共 $n!$ 项。a_{ij} 表示第 i 行 j 列的元素。

MATLAB 是采用矩阵行列式函数 det 来计算行列式，若 n 阶方阵为 $\boldsymbol{A}$，则 d = det（A）返回方阵 $\boldsymbol{A}$ 的行列式。假如矩阵 $\boldsymbol{A}$ 的元素是整数，则它的行列式也是整数。行列式的计算是从高斯消去法得到 LU 分解，即

```
[l,u] = lu(A)                  % 分解成上三角矩阵 u 和下三角矩阵 l，详见
                                 2.35 节矩阵的 LU 分解
s = det(l)                     %始终是 1 或 -1
det(A) = s* prod((diag(u))     %计算上三角矩阵对角元素的乘积，再乘以正号或
                                负号。diag（u）为矩阵 u 的对角线向量，详见
                                2.28 节对角矩阵。prod 为计算向量元素的乘积，
                                详见 7.3 节数组元素的乘积
```

【例 2-12】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 10\end{bmatrix}$，求矩阵 $\boldsymbol{A}$ 的行列式。

解：

```
≫A = [1 2 3;4 5 6;7 8 10]                    %设 A 为已知矩阵
A =

     1        2        3
     4        5        6
     7        8       10

≫det(A)                                       %矩阵 A 的行列式值
ans =
    -3
```

2.11 矩阵的转置

矩阵的行和列对换称为矩阵的转置，对矩阵的转置运算，只需在矩阵的右上角加上单引号即可

【例 2-13】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$，求矩阵 $\boldsymbol{A}$ 的转置矩阵。

解：

```
≫A = [1 2 3;4 5 6;7 8 9]                      %设 A 为已知矩阵
A =

     1        2        3
     4        5        6
     7        8       10

≫ A'                                          %矩阵 A 的转置，即对应的行与列对换
ans =

     1        4        7
     2        5        8
     3        6       10
```

2.12 单位矩阵

单位矩阵为主对角线元素全为 1，而其他元素全为零的矩阵，数学上以 $\boldsymbol{I}$ 表示之。MATLAB 则以 eye（m，n）表示，其中 m 为行数，n 为列数，若为方阵，则以 eye（n）表示。

【例 2-14】 求 4 行 4 列的单位矩阵。

解：

```
≫eye(4)
ans =
```

```
        1         0         0         0
        0         1         0         0
        0         0         1         0
        0         0         0         1
```

【例 2-15】 求 3 行 4 列的单位矩阵。

解：

```
≫eye(3,4)
ans =
        1      0      0      0
        0      1      0      0
        0      0      1      0
```

2.13 全 1 矩阵

矩阵中所有元素为 1，称为全 1 矩阵，全 1 矩阵以 ones（m,n)表示，其中 m 为行数，n 为列数。若为方阵则以 ones（n）表示。

【例 2-16】 求 4 阶全 1 矩阵。

解：

```
≫ones(4)
ans =
        1      1      1      1
        1      1      1      1
        1      1      1      1
        1      1      1      1
```

【例 2-17】 求 3 行 4 列的全 1 矩阵。

解：

```
≫A = ones(3,4)
A =
        1      1      1      1
        1      1      1      1
        1      1      1      1
```

2.14 零矩阵

零矩阵为矩阵内所有元素为零。通常首先用来确定矩阵的大小，随后为矩阵元素赋值。零矩阵以 zeros (m,n)表示，其中 m 为行数，n 为列数，若为方阵，则以 ones (n)表示。

【例 2-18】 求 4 阶零矩阵。

解：

```
≫zeros(4)
ans =
        0      0      0      0
```

```
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

【例 2-19】 求 3 行 4 列的零矩阵。

解：

```
>>zeros(3,4)
ans =
     0     0     0     0
     0     0     0     0
     0     0     0     0
```

2.15 魔方矩阵

魔方矩阵是 $n \times n$ 元素所构成的方阵，其每个元素由不同的 $1 \sim n^2$ 的整数所组成，它的每行、每列以及对角线元素之和均相等，并等于n $(1+n^2)$ $/2$ 。魔方矩阵以 magic（n）表示之，其中 n 为矩阵的阶次。对于魔方矩阵的程序，MATLAB 是隐藏的。在第 3 章将介绍奇阶魔方的程序是如何编制的。

【例 2-20】 求 4 阶魔方矩阵。

解：

```
>>A = magic(4)                %4 阶魔方矩阵。其中每行、每列之和以及每根
                                对角线元素的和均为 34,
                                magic 为魔方矩阵的调用函数
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
>>sum(A,1)                    %检查矩阵 A 各列元素之和，均为 34。sum(A,
                                1)为求矩阵 A 各列元素之和
ans =
    34    34    34    34
>>sum(A,2)                    %检查矩阵 A 各行元素之和，均为 34。sum(A,
                                2)为求矩阵 A 各行元素之和
ans =
    34
    34
    34
    34
>>trace(A)                    %检查矩阵 A 主对角线元素之和为 34。trace 为
                                求主对角线元素之和的函数
```

```
ans =
    34
≫trace(fliplr(A))                    %检查矩阵 A 副对角线元素之和为 34
ans =                                  fliplr 为矩阵左右翻转函数，详见 2.27 节
    34
```

2.16 Pascal 矩阵

Pascal 矩阵是一个实对称的正定矩阵，它由 Pascal 三角形组成，Pascal 三角形是由 0 到 $2n-1$ 阶的二项式系数组成，二项式系数见表 2-2，把二项式系数依此填写在矩阵的左侧对角线上，提取左侧的 n 行和 n 列即为 Pascal 矩阵。图 2-1 所示即为 4 阶 Pascal 矩阵。

表 2-2　二项式系数

阶　　数	二项式系数
0	1
1	1 1
2	1 2 1
3	1 3 3 1
4	1 4 6 4 1
5	1 5 10 10 5 1
6	1 6 15 20 15 6 1
7	1 7 21 35 35 21 7 1
…	…
n	1 n n（$n-1$）/2！ … n 1

1	1	1	1	1	1	1
1	2	3	4	5	6	
1	3	6	10	15		
1	4	10	20			
1	5	15				
1	6					
1						

图 2-1　由 Pascal 三角形生成 pascal（n）的示意图

对于 n 阶 Pascal 矩阵的书写格式为

A = pascal(n)

A1 = pascal(n,1)

A2 = pascal(n,2)

A = pascal（n），它返回一个对称的正定的矩阵，它的每一个元素有以下特点：

$A(i,j)=A(i-1,j)+A(i,j-1)$，且 $A(i,1)=1$；$A(1,j)=1$，其中 i、$j=1\sim n$

Pascal（n）的每一个元素为整数，它的逆矩阵的所有元素也为整数。

对于模式 1 的 Pascal 矩阵，A1 = pascal（n,1）它返回下三角的 Cholesky 因式，而且它列的符号是正负交替的。它看上去是杂乱的，但它的逆矩阵恰好等于它自己。

对于模式 2 的 Pascal 矩阵，A2 = pascal (n,2)它的立方恰好是 eye (n)矩阵。它与 A1 的关系是

A2 = fliplr(A1')*(-1)^(n-1)

式中，fliplr 为矩阵左右翻转函数，详见 2.27 节。

【例 2-21】 求 5 阶 Pascal 矩阵。

解：

```
>>pascal(5)                    %5 阶 Pascal 矩阵
ans =

     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15
     1     4    10    20    35
     1     5    15    35    70
```

【例 2-22】 求模式 1 的 4 阶 Pascal 矩阵 $\boldsymbol{A}_1$ 和模式 2 的 Pascal 矩阵 $\boldsymbol{A}_2$，并验证 $\boldsymbol{A}_2$ 是 $\boldsymbol{A}_1$ 经转置，并左右翻转后乘以$(-1)^{n+1}$，其中 n 为矩阵的阶数。

解：

```
>>A1 = pascal(4,1)             %模式 1 的 Pascal 矩阵
A1 =

     1     0     0     0
     1    -1     0     0
     1    -2     1     0
     1    -3     3    -1

>>A2 = pascal(4,2)             %模式 2 的 Pascal 矩阵
A2 =

    -1    -1    -1    -1
     3     2     1     0
    -3    -1     0     0
     1     0     0     0

>>B = -fliplr(A1')             %将矩阵 A1 转置、翻转再取负，即成矩阵 A2
C =

    -1    -1    -1    -1
     3     2     1     0
    -3    -1     0     0
     1     0     0     0
```

【例 2-23】 验证 Pascal (6,2)^3 = eye(6)

证：

```
>>A = pascal(6,2)           %6 阶模式 2 Pascal 矩阵
```

```
A =
    -1      -1      -1      -1      -1      -1
     5       4       3       2       1       0
   -10      -6      -3      -1       0       0
    10       4       1       0       0       0
    -5      -1       0       0       0       0
     1       0       0       0       0       0
>>A^3                      %山此证明 A^3 = eye (6)，可以证明对任意阶次均成立
ans =
    1    0    0    0    0    0
    0    1    0    0    0    0
    0    0    1    0    0    0
    0    0    0    1    0    0
    0    0    0    0    1    0
    0    0    0    0    0    1
```

【例 2-24】 验证 5 阶 Pascal 矩阵的逆矩阵的元素为整数。

证明：

```
>>A = pascal(5)            %5 阶 Pascal 矩阵
A =
    1       1       1       1       1
    1       2       3       4       5
    1       3       6      10      15
    1       4      10      20      35
    1       5      15      35      70
>>inv(A)                   %矩阵 A 的逆矩阵元素，确为整数。可以证明对任意阶次的 Pas-
                             cal 矩阵，它的逆矩阵元素均为整数
ans =
     5     -10      10      -5       1
   -10      30     -35      19      -4
    10     -35      46     -27       6
    -5      19     -27      17      -4
     1      -4       6      -4       1
```

2.17 Hilbert 矩阵

Hilbert 矩阵的每个元素的值，山行数 i 和列数 j 决定，等于$1/(i+j-1)$，所以特别容易用编程来列写，但它是坏的条件数矩阵的例子。Hilbert 矩阵以 Hilb (m，n)C 表示之，其中 m 为行数，n 为列数，若为方阵，则以 Hilb (n)表示。

【例 2-25】 求 4 阶 Hilbert 矩阵，分别用小数和分数表示。

解：

```
≫Hilb(4)                    %4 阶 Hilbert 矩阵，以小数表示
ans =

    1.0000      0.5000      0.3333      0.2500
    0.5000      0.3333      0.2500      0.2000
    0.3333      0.2500      0.2000      0.1667
    0.2500      0.2000      0.1667      0.1429

≫format rat                 %设置输出格式为分数
≫A = hilb(4)                %4 阶 Hibert 矩阵，赋值给予 A
A =                         %分数格式显示

     1      1/2     1/3     1/4
    1/2     1/3     1/4     1/5
    1/3     1/4     1/5     1/6
    1/4     1/5     1/6     1/7
```

2.18 均匀分布的随机矩阵

rand (m，n)是一个用随机数输入的 m × n 的矩阵，随机数是选择均匀分布在（0，1）区间内。随机数的产生，是由正规的伪随机数发生器产生。当 MATLAB 调用 rand，则随机数发生器复位，并产生机会均等的而不受状态改变的随机数。

【例 2-26】 求两个 4 阶均匀分布的随机矩阵，分别赋值给 $\boldsymbol{A}_1$、$\boldsymbol{A}_2$。

解：

```
≫A1 = rand(4)               %4 阶随机矩阵
A1 =

    0.9501      0.8913      0.8214      0.9218
    0.2311      0.7621      0.4447      0.7382
    0.6068      0.4565      0.6154      0.1763
    0.4860      0.0185      0.7919      0.4057

≫A2 = rand(4)
A2 =

    0.9355      0.0579      0.1389      0.2722
    0.9169      0.3529      0.2028      0.1988
    0.4103      0.8132      0.1987      0.0153
    0.8936      0.0099      0.6038      0.7468
```

2.19 正态分布的随机矩阵

randn (m，n)产生以正态分布的随机矩阵，其中 m 为行数，n 为列数，若为方阵，则以 randn (n)表示。

【例 2-27】 试列出 4 阶的正态分布的随机矩阵。

解:

```
randn(4)
ans =

   -0.4326    -1.1465     0.3273    -0.5883
   -1.6656     1.1909     0.1746     2.1832
    0.1253     1.1892    -0.1867    -0.1364
    0.2877    -0.0376     0.7258     0.1139
```

2.20 矩阵的大小

size 命令用来测试矩阵的大小，对于 $m \times n$ 的矩阵 $\boldsymbol{x}$，size (x)返回一个行向量 $\boldsymbol{d}=[m, n]$，它包含了矩阵的行数 m 和列数 n。如果专为显示矩阵的行数和列数，则可用以下显示格式：

m = size(x,1)　　n = size(x,2)

对于多维矩阵 x，size (x)返回一个行向量 $\boldsymbol{d}=[m\ n\ p \cdots s]$，它显示矩阵的行数 m、列数 n、页数 p 和多重页数 s。size 常用于构造一个与已知矩阵相同大小的新矩阵。下面举例说明。

【例 2-28】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 1 & 1 & 1 & 1\\ 1 & 2 & 3 & 4 & 5\\ 1 & 3 & 6 & 10 & 15\end{bmatrix}$，求矩阵的大小。

解:

```
>>A = [1 1 1 1 1;1 2 3 4 5;1 3 6 10 15]   %设 A 为已知矩阵
A =

     1     1     1     1     1
     1     2     3     4     5
     1     3     6    10    15

>>d = size(A)                             %测试矩阵 A 的尺寸
d =                                       %矩阵 A 为 3 行 5 列

     3     5
```

【例 2-29】 已知 A(:,:,1) = magic(3), A(:,:,2) = pascal(3), A(:,:,3) = zeros(3), A(:,:,4) = ones(3),(其中冒号“:”表示所有行或所有列)，求 size (A)。

解:

```
>>A(:,:,1) = magic(3);A(:,:,2) = pascal(3);A(:,:,3) = zeros(3);A(:,:,4) = ones(3);
                                          %设 3 维矩阵 A 的第 1 页为 magic (3),
                                          %设 3 维矩阵 A 的第 2 页为 pascal (3),
                                          %设 3 维矩阵 A 的第 3 页为 zeros (3),
                                          %设 3 维矩阵 A 的第 4 页为 ones (3)
>> A                                      %显示矩阵 A
A(:,:,1) =                                %矩阵 A 的第 1 页
```

```
       8         1         6
       3         5         7
       4         9         2
A(:,:,2) =                              %矩阵 A 的第 2 页
       1         1         1
       1         2         3
       1         3         6
A(:,:,3) =                              %矩阵 A 的第 3 页
       0         0         0
       0         0         0
       0         0         0
A(:,:,4) =                              %矩阵 A 的第 4 页
       1         1         1
       1         1         1
       1         1         1
>>size(A)                               %测试矩阵 A 的大小
ans =                                   %矩阵 A 为 3 行 3 列 4 页
       3         3         4
```

【例 2-30】 已知矩阵 ***A*** 为 4 阶 Pascal 矩阵，求制作一个与矩阵 ***A*** 同阶的单位矩阵 ***B***。

解：

```
>>A = pascal(4)                         %已知矩阵 A
A =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20

>>B = eye(size(A))                      %与矩阵 A 同阶的单位矩阵 B
B =


     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

2.21 矩阵的秩

矩阵的秩是对矩阵行（或列）线性不相关数的评估。所谓线性不相关，是指某行（或某列）不能由其他行（或列）以线性组合表达。若已知矩阵 ***A***，则矩阵的秩可用 rank（A）求得。

【例 2-31】 求矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 10\end{bmatrix}$的秩。

解：

```
≫A = [1 2 3;4 5 6;7 8 10]          %已知矩阵 A
A =

     1     2     3
     4     5     6
     7     8    10

≫rank(A)                           %矩阵 A 的秩为 3，是满秩。由此可见 A 的行和列是彼
                                      此独立的
ans =
     3
```

【例 2-32】 求矩阵 $\boldsymbol{B}=\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\\7 & 8 & 9\end{bmatrix}$的秩。

解：

```
≫B = [1 2 3;4 5 6;7 8 9]           %已知矩阵 B
B =
     1     2     3
     4     5     6
     7     8     9
≫rank(B)                           %矩阵 B 的秩为 2，非满秩。因 B 的第 3 行可由第 1、2
                                      行线性组合表示，如下列程序所示
ans =
     2
≫B(1,:)                            %B 的第 1 行
ans =
     1     2     3
≫B(2,:)                            %B 的第 2 行
ans =
     4     5     6
≫B(2,:)*2 - B(1,:)C                %B 的第 1、2 行的线性组合等于第 3 行，所以第 3 行不是
                                      独立的
ans =
     7     8     9
```

2.22 向量的范数

向量的范数是表示向量 $\boldsymbol{V}$ 特征的量，它的 p 范数定义为

$$\| \boldsymbol{V} \|_p = \left[\sum_{i=1}^{n} | v_i |^p \right]^{1/p} \tag{2-3}$$

式中，$\boldsymbol{V}$ 为向量；v_i 为向量 $\boldsymbol{V}$ 中的元素；n 为元素数。

当 p 具有 1、2、inf 和 – inf 等特殊值时，范数的计算公式和 MATLAB 书写格式见表 2-3。

表 2-3　范数计算公式和 MATLAB 书写格式

p	向量范数计算公式	MATLAB 书写格式
1	$\sum_{i=1}^{n} \| v_i \|$	norm(v,1)
2	$\sqrt{\sum_{i=1}^{n} \| v_i \|^2}$	norm（v，2）或 norm（v）
– inf	$\min \sum_{i=1}^{n} \| v_i \|$	norm(v，– inf)
+ inf	$\max \sum_{i=1}^{n} \| v_i \|$	norm(v，+ inf)

范数可以理解为向量的长度，它对于分析参数变化对线性方程组解的灵敏度影响是有用的。

【例 2-33】　已知向量 $\boldsymbol{v}$ = [1 2 3 4]，求 p = 1、2、inf、 – inf 时向量 $\boldsymbol{v}$ 的范数。

解：

```
≫v = [1 2 3 4];              %输入向量 v
≫norm(v,1)                   %1 范数
ans =
      10
≫norm(v,2)                   %2 范数
ans =
      5.4772
≫norm(v, + inf)              % + inf 范数
ans =
      4
≫norm(v, - inf)              % - inf 范数
ans =
      1
```

2.23　矩阵的范数

矩阵的范数是从向量的范数引伸出来的，同样具有长度的意义。
它的书写格式为

n = norm(A)

n = morm(A,p)

对于形式 n = norm（A），它返回一个标量，这个标量是矩阵 A 的最大奇异值。有关奇异

值，详见 2.25 节。

对于形式 n = norm（A，p），它返回不同类型的范数，决定于 p 的值，见表 2-4。

它的 1、2、inf 和 Frobenius 范数的定义见表 2-4。其中 $a_{i,j}$ 为矩阵 $\boldsymbol{A}$ 的 i 行、j 列的元素。

表 2-4　矩阵范数计算公式和 MATLAB 书写格式

p	矩阵范数计算公式	MATLAB 书写格式
1	$\max\limits_{0 \leqslant j \leqslant n} \sum_{i=1}^{m} \lvert a_{i,j} \rvert$	norm(A,1) = max(sum(abs(A)))
2	max (svd (A))①	norm(A,2) or norm(A)
inf	$\max\limits_{0 \leqslant i \leqslant m} \sum_{j=1}^{n} \lvert a_{i,j} \rvert$	norm(A,inf) = max(sum(abs(A')))
Frobenius 范数	$\left[\sum_{i=1}^{n}\sum_{j=1}^{m} \lvert a_{i,j} \rvert^{2}\right]^{1/2}$	norm(A,'fro') = sqrt(sum(diag(A'*A)))

① 表中 svd 为奇异值分解，详见 2.25 节。

【例 2-34】　已知 $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 5 & 8 \\ 1 & 3 & 5 & 7 \\ 3 & 4 & 7 & 11 \end{bmatrix}$，求 1、2、inf 阶和 Frobenius 范数。

解：

```
>>A = [1 2 3 4; 2 3 5 8;1 3 5 7;3 4 7 11]    %已知矩阵 A
A =

     1     2     3     4
     2     3     5     8
     1     3     5     7
     3     4     7    11

>>A1 = norm(A,1)                             %矩阵 A 的 1 阶范数
A1 =
    30
>>max(sum(A))                                %矩阵 A 的列和最大值，等于范数 1。max 为求
                                              最大值函数，sum 为求矩阵列元素和的函数
ans =
    30
>>Ai = norm(A,inf)                           %矩阵 A 的 inf 阶范数
Ai =
    25
>>max(sum(A'))                               %矩阵的行和最大值，等于 inf 阶范数
ans =
    25
```

```
≫A2 = norm(A,2)                          %矩阵的范数 2
A2 =
    20.2435
≫max(svd(A))                             %矩阵 A 的奇异值最大值，等于范数 2。有关奇
                                          异值分解 svd 见 2.25 节
ans =
    20.2435
    ≫Af = norm(A,'fro')                   %矩阵 A 的 Frobenius 范数
Af =
    20.2731
≫sum(abs(A(1:16)).^2).^(1/2)             %矩阵 A 的元素平方和再开方，等于 Frobenius 范
                                          数。这是用范数计算公式计算
ans =
    20.2731
≫sqrt(sum(diag(A'*A)))                   % 用 MATLAB 公式计算。diag(A'*A)，为矩
                                          阵 A 所有元素的二次方和
ans =
    20.2731
```

2.24 矩阵的条件数

矩阵的条件数是一个与逆矩阵inv(x)有关的数，其中 x 为矩阵。它的书写格式为

cond(x)

cond(x,p)

矩阵的条件数是用来测量线性方程组的解对数据输入误差的灵敏度。它从线性方程组的逆矩阵中给出一个解的精度指示。cond(x)和 cond(x,p)接近于 1，则表示这个矩阵有良好的条件数。条件数越大，则表示矩阵越接近奇异。矩阵条件数cond(x)返回矩阵 x 的 2 范数与逆矩阵inv(x)的2 范数之积，即

cond(x) = norm(x) * norm(inv(x))

或者　　cond(x,p) = norm(x,p) * norm(inv(x),p)

p 取值为 1、2、inf 和 fro。对于 2 范数的条件数，还可能从矩阵 x 的最大奇异值与最小奇异值之比来求取。

【例 2-35】 计算矩阵 $\boldsymbol{A}=\begin{bmatrix}1&3&4&5\\1&1&3&4\\1&1&1&3\\1&1&1&1\end{bmatrix}$的条件数。

解：

```
≫A = [1 3 4 5;1 1 3 4;1 1 1 3;1 1 1 1]   %输入矩阵 A
A =
```

```
     1     3     4     5
     1     1     3     4
     1     1     1     3
     1     1     1     1
>>inv(A)                              %计算逆矩阵 A
ans =
   -0.5000    0.2500    0.1250    1.1250
    0.5000   -0.7500    0.1250    0.1250
         0    0.5000   -0.7500    0.2500
         0         0    0.5000   -0.5000
>>C1 = norm(A)* norm(ans)             %用公式计算矩阵 A 的条件数
C1 =
   13.9424
>>C = cond(A)                         %直接调用条件数函数 cond
C =
   13.9424
>>s = svd(A)                          %计算矩阵 A 的奇异值
s =
   9.5410
   1.2253
   1.0000
   0.6843
>>C2 = max(s)/min(s)                  % 用奇异值的最大值与最小值之比计算条件数,
                                        结果完全一致
C2 =
   13.9424
```

2.25 矩阵的奇异值和奇异值分解

矩阵 $\boldsymbol{A}$ 的奇异值返回一个奇异值列向量 $\boldsymbol{s}$，用s = svd(A)表示。

矩阵 $\boldsymbol{A}$ 的奇异值分解，则返回一个与矩阵 $\boldsymbol{A}$ 大小相同的对角矩阵 $\boldsymbol{s}$ 和二个酉矩阵 $\boldsymbol{u}$，$\boldsymbol{v}$，且满足 $\boldsymbol{A} = \boldsymbol{u} * \boldsymbol{s} * \boldsymbol{v}$，若 $\boldsymbol{A}$ 为 $m \times n$ 阵，则 $\boldsymbol{u}$ 为 $m \times m$，$\boldsymbol{v}$ 为 $n \times n$，奇异值在 $\boldsymbol{s}$ 主对角线上，且为非负降序排列。所谓酉矩阵是这样的矩阵，它的逆矩阵等于它的共轭转置矩阵。

【例 2-36】 求 $\boldsymbol{A} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$的奇异值和奇异值分解。

解:

```
>>A = [1,2,3;4,5,6;7,8,9]             %已知矩阵 A
```

```
A =

     1     2     3
     4     5     6
     7     8     9

>>svd(A)                                 %矩阵 A 的奇异值
ans =

   16.8481
    1.0684
    0.0000

>>[u,s,v] = svd(A)                       %矩阵 A 的奇异值分解
u =

   -0.2148    0.8872    0.4082
   -0.5206    0.2496   -0.8165
   -0.8263   -0.3879    0.4082

s =

   16.8481         0         0
         0    1.0684         0
         0         0    0.0000

v =

   -0.4797   -0.7767   -0.4082
   -0.5724   -0.0757    0.8165
   -0.6651    0.6253   -0.4082

>>iu = inv(u)                            %核对矩阵 u 是否是酉矩阵
iu =

   -0.2148   -0.5206   -0.8263
    0.8872    0.2496   -0.3879
    0.4082   -0.8165    0.4082

>>u'                                     %逆矩阵 iu 等于 u'，所以 u 是酉矩阵
ans =

   -0.2148   -0.5206   -0.8263
    0.8872    0.2496   -0.3879
    0.4082   -0.8165    0.4082

>>iv = inv(v)                            %核对矩阵 v 是否是酉矩阵
iv =

   -0.4797   -0.5724   -0.6651
```

```
    -0.7767      -0.0757       0.6253
    -0.4082       0.8165      -0.4082
>>v'                                    %逆矩阵 iv 等于 v'，所以 v 是酉矩阵
ans =
    -0.4797      -0.5724      -0.6651
    -0.7767      -0.0757       0.6253
    -0.4082       0.8165      -0.4082
```

2.26 矩阵的特征值和特征向量

对于 n 阶方阵 $\boldsymbol{A}$，所组成的特征方程 $f(\lambda) = \det|(\boldsymbol{A} - \lambda \boldsymbol{I})| = 0$ 的全部根，它就是方阵 $\boldsymbol{A}$ 的特征值。用向量d = eig(A)表示。

对于 n 阶方阵 $\boldsymbol{A}$，如果存在一个以特征值组成的对角矩阵 $\boldsymbol{d}$ 和 $\boldsymbol{n}$ 阶矩阵 $\boldsymbol{v}$，使 $\boldsymbol{Av} = \boldsymbol{vd}$，矩阵 $\boldsymbol{v}$ 的每一个列向量对应于特征值的特征向量。用[v,d] = eig(A)表示。

【例 2-37】 已知方阵 $\boldsymbol{A} = \begin{bmatrix} -6 & -11 & -6 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$，求其特征值。

解：

```
>>A = [-6 -11 -6;1 0 0;0 1 0]           %已知矩阵 A
A =
     -6     -11     -6
      1       0      0
      0       1      0
>>syms lambda                           %syms 为快捷创建符号变量，lambda 为符号变量
                                          名
>>det(A - lambda* eye(3))               %列写方阵 A 的特征方程式
ans =
-6* lambda^2 - lambda^3 - 11* lambda - 6
>>P = sym2poly(ans)                     %转换符号多项式方程为系数多项式向量，sym2poly
                                          为转换函数
P =
     -1       -6       -11       -6
>>roots(P)                              %解多项式的根，roots 为多项式求根函数
ans =
     -3.0000
     -2.0000
     -1.0000
>>d = eig(A)                            %eig 为 MATLAB 求解特征值函数，计算结果与矩阵
```

```
                                                    A所构成的特征方程式的特征根相同
d =
    -3.0000
    -2.0000
    -1.0000
```

【例 2-38】 已知 4 阶魔方矩阵 A，求其特征值和特征向量。

解：

```
≫A = magic(4)
A =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
≫d = eig(A)                              %矩阵 A 的特征值
d =
   34.0000
    8.9443
   -8.9443
    0.0000
≫[V,D] = eig(A)                          %矩阵 A 的特征向量所组成矩阵 V
V =
   -0.5000   -0.8236    0.3764   -0.2236
   -0.5000    0.4236    0.0236   -0.6708
   -0.5000    0.0236    0.4236    0.6708
   -0.5000    0.3764   -0.8236    0.2236
D =
   34.0000         0         0         0
         0    8.9443         0         0
         0         0   -8.9443         0
         0         0         0    0.0000
```

2.27 矩阵的左右翻转、上下翻转和矩阵的逆时针旋转 90°操作

为了方便改写矩阵，MATLAB 设置了矩阵的左右翻转、上下翻转和矩阵的逆时针旋转 90°的操作命令 fliplr、flipud 和 rot90。

【例 2-39】 将 3 阶魔方阵中的列从左向右翻转。

解：

```
≫A = magic(3)                            %输入已知矩阵 A
A =
```

```
    8     1     6
    3     5     7
    4     9     2
≫fliplr(A)                    %将矩阵 A 的列左右翻转，fliplr 为矩阵左右翻转的函数
ans =
    6     1     8
    7     5     3
    2     9     4
```

【例 2-40】 将矩阵 3 阶 pascal 阵的行上、下翻转。

解：

```
≫A = pascal(3)                % 设 A 为pascal(3)矩阵
A =
    1     1     1
    1     2     3
    1     3     6
≫flipud(A)                    %矩阵 A 的行上、下翻转
ans =
    1     3     6
    1     2     3
    1     1     1
```

【例 2-41】 将矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2 & 3 & 4\\ 5 & 6 & 7 & 8\end{bmatrix}$ 逆时针旋转 90°。

解：

```
≫A = [1 2 3 4;5 6 7 8]        %输入矩阵 A
A =
    1     2     3     4
    5     6     7     8
≫rot90(A)                     %矩阵 A 逆时针旋转 90°
ans =
    4     8
    3     7
    2     6
    1     5
≫A'                           %注意，旋转 90°操作与矩阵 A 转置不同
ans =
```

```
     1     5
     2     6
     3     7
     4     8
≫B = flipud(rot90(A))              %矩阵 A 旋转 90°以后，再进行上、下翻转，则与
                                    矩阵 A 转置相同，即 B = A′
B =
     1     5
     2     6
     3     7
     4     8
```

2.28 对角矩阵

对角矩阵函数 diag 有双重意义。它的书写格式之一为

X = diag(v,k)

将向量 v 写入矩阵 X 的主对角线上，而矩阵 X 的其他元素为零。k 表示上移或下移行数，正表示上移，负表示下移，k = 0 则恰好在主对角线上，当 k = 0 时可以默认不写。

另一种书写格式为

v = diag(X,k)

它是从矩阵 X 中提取对角线元素到向量 v 上。k = 0 或默认，则提取主对角线元素，否则提取上移 k 行或下移 k 行的对角线元素，正号表示上移，负号表示下移。

【例 2-42】 已知行向量 $\boldsymbol{v}$ = [1 2 3 4]，将 $\boldsymbol{v}$ 向量元素写入矩阵的主对角线，求对角矩阵，以及上移一行的对角矩阵和下移一行的对角矩阵。

解：

```
≫v = [1 2 3 4]                     %已知行向量 v，即已知对角矩阵的元素
v =
    1     2     3     4
≫diag(v)                           %对角矩阵命令，亦可写作diag(v,0)
ans =
    1     0     0     0
    0     2     0     0
    0     0     3     0
    0     0     0     4
≫diag(A,1)                         %对角矩阵上移一行，但阶数变为 5 阶
ans =
    0     1     0     0     0
    0     0     2     0     0
    0     0     0     3     0
```

```
     0     0     0     0     4
     0     0     0     0     0
>>diag(A,-1)                  %对角矩阵下移一行，但阶数变为5阶
ans =
     0     0     0     0     0
     1     0     0     0     0
     0     2     0     0     0
     0     0     3     0     0
     0     0     0     4     0
```

【例 2-43】 已知 $\boldsymbol{u}=$ [2 3 5 8 11]，创建 Vandermonde 矩阵 $\boldsymbol{X}$，提取 $\boldsymbol{X}$ 的主对角线赋予向量 v，主对角线上移一行赋予向量 v_1，下移一行赋予 v_{n1}。

解：

```
>>u=[2 3 5 8 11];             %输入向量u
>>X=vander(u)                 %创建Vandermonde矩阵X
X =

      16        8        4        2        1
      81       27        9        3        1
     625      125       25        5        1
    4096      512       64        8        1
   14641     1331      121       11        1

>>v=diag(X)'                  %提取X的主对角线，并转置
v =

    16    27    25     8     1

>>v1=diag(X,1)'               %提取X的主对角线的上一行，并转置
v1 =

     8     9     5     1

>>vn1=diag(X,-1)'             %提取X的主对角线的下一行，并转置
vn1 =

    81   125    64    11
```

2.29 矩阵的重组 1

为了将已知矩阵重新组合或改写，设置了各种简便的方法将矩阵重组，因此可以减少矩阵元素的重复输入。下面介绍用新的行或列取代原有矩阵的行或列。

【例 2-44】 已知向量 $\boldsymbol{A}$，将向量 $\boldsymbol{A}$ 取代矩阵 $\boldsymbol{B}$ 的某一行或矩阵 $\boldsymbol{C}$ 的某一列。

解：

```
>>A=[1 2 3 4]                 %设已知向量A
A =
```

```
     1     2     3     4
>>B = pascal(4)                      % 不失一般性，设矩阵 B 为Pascal(4)矩阵
B =
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
>>B(1,:) = A                         % 将矩阵 B 的第一行换成行向量 A，冒号表示各
                                       列。注意向量 A 的列数必须与B(1,:)的列数相
                                       等
B =                                  % 替换后的矩阵 B
     1     2     3     4
     1     2     3     4
     1     3     6    10
     1     4    10    20
>>C = pascal(4)                      % 重新设置 4 阶矩阵 C
C =
     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
>>C(:,1) = A'                        % 将矩阵 C 的第一列换成列向量 A'，冒号表示各
                                       行。注意向量 A'，必须与C(:,1)的行数相等
C =                                  % 替换后的矩阵 C
     1     1     1     1
     2     2     3     4
     3     3     6    10
     4     4    10    20
```

2.30　矩阵的重组 2

对矩阵的 2 行或 2 列进行对换操作。

【例 2-45】　已知矩阵 A，将其第一行与第四行对换。

解：

```
>>A = magic(4)                       % 不失一般性，令 A 为 4 阶魔方矩阵
A =
    16     2     3    13
     5    11    10     8
```

```
     9     7     6    12
     4    14    15     1
>>c = A(1,:)                     %取 A 的第 1 行，赋值给向量 c
c =
    16     2     3    13
>>A(1,:) = A(4,:)                %取 A 的第 4 行，赋值给矩阵 A 的第 1 行
A =
     4    14    15     1
     5    11    10     8
     9     7     6    12
     4    14    15     1
>>A(4,:) = c                     %将向量 c 赋值给矩阵 A 的第 4 行，2 行对换完成
A =
     4    14    15     1
     5    11    10     8
     9     7     6    12
    16     2     3    13
```

2.31 矩阵的重组 3

从矩阵中选取子矩阵。

【例 2-46】 已知 5 阶魔方矩阵，取其中的前 4 阶组成矩阵 B。

解：

```
>>A = magic(5)                   %不失一般性，设 A 为 5 阶魔方矩阵
A =
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
>>B = A(1:4,1:4)                 % 取 A 矩阵的前 4 行、4 列即为所求，这里冒号
                                   表示“到”的意思
B =
    17    24     1     8
    23     5     7    14
     4     6    13    20
    10    12    19    21
```

【例 2-47】 已知 4 阶矩阵 ***A***，取其第 2 行至第 4 行、第 2 列至第 4 列所构成的 3 阶

矩阵。

解：

```
≫A=[1 3 5 7;2 3 5 8;2 4 6 8;3 4 7 11]   %设A为已知矩阵
A=
     1     3     5     7
     2     3     5     8
     2     4     6     8
     3     4     7    11
≫B=A(2:4,2:4)                           %将矩阵A的2到4行，2到4列赋值给矩阵B，
                                          即为所求
B=
     3     5     8
     4     6     8
     4     7    11
```

【例2-48】 已知4阶矩阵 ***A*** 同上题，求2行2列对应元素的子矩阵。

解：

```
≫A=[1 3 5 7;2 3 5 8;2 4 6 8;3 4 7 11]   %设A为已知矩阵
A=
     1     3     5     7
     2     3     5     8
     2     4     6     8
     3     4     7    11
≫A(2,:)=[],A(:,2)=[]                   %令第2行为空行，再令第2列为空列，
A=                                        第2行被清空
     1     3     5     7
     2     4     6     8
     3     4     7    11
A=                                       %第2列被清空，子矩阵列写完成
     1     5     7
     2     6     8
     3     7    11
```

2.32 矩阵的重组4

将矩阵改写成行向量或列向量。

【例2-49】 将已知3阶魔方阵 ***A*** 矩阵的所有元素按列改写成行向量 ***B***。

解：

```
≫A=magic(3)                  %设A为3阶魔方矩阵
```

```
A =

     8     1     6
     3     5     7
     4     9     2

>>B = A(:)'                    %按列展开，并排成行向量B
  B =

     8     3     4     1     5     9     6     7     2
```

2.33 矩阵的重组 5

用 reshape 命令改变矩阵的大小。

【例 2-50】 将 4 阶 Hilbert 矩阵 $\boldsymbol{A}$，改写成 2 行 8 列的矩阵 $\boldsymbol{B}$，并以分数表示。

解：

```
>>format rat                   %设定分数显示格式
>>A = hilb(4)                  %将4阶Hilbert矩阵赋值给A
A =

     1      1/2    1/3    1/4
    1/2     1/3    1/4    1/5
    1/3     1/4    1/5    1/6
    1/4     1/5    1/6    1/7

>>B = reshape(A,2,8)           %改变矩阵尺寸成2行8列，reshape为改变矩阵尺寸命令。
                                注意，采用reshape命令时，改写前后的矩阵元素必须相
                                等，否则显示出错
B =
  Columns 1 through 6

     1      1/3    1/2    1/4       1/3    1/5
    1/2     1/4    1/3    1/5       1/4    1/6

  Columns 7 through 8

    1/4     1/6
    1/5     1/7
```

2.34 逆矩阵

对于 n 阶方阵 $\boldsymbol{A}$，如果存在 $\boldsymbol{AB}=\boldsymbol{BA}=\boldsymbol{I}$（单位矩阵），则称 $\boldsymbol{B}$ 为 $\boldsymbol{A}$ 的逆矩阵。逆矩阵存在的必要和充分条件为 $\boldsymbol{A}$ 的行列式不等于零。在 MATLAB 中，若方阵为 A，则逆矩阵的函数为inv(A)，也可以用A \eye(n)，eye(n)/A，A^(－1)，rref([A，eye(size(A))])及逆矩阵计算公式来求得。逆矩阵广泛用于解线性方程组的求解。

这几种计算方法，在矩阵 $\boldsymbol{A}$ 非奇异的条件下，计算结果都是相同的。但在采用左除法时，运算速度较快，对于以消元法为基础的 rref 函数，在主元元素接近为零时，则舍入误差

较大。

【例 2-51】 已知 4 阶方阵 $A=\begin{bmatrix}1&2&3&4\\2&3&5&8\\1&3&5&7\\3&4&7&11\end{bmatrix}$，求逆矩阵。

解：

```
≫A=[1 2 3 4;2 3 5 8;1 3 5 7;3 4 7 11]    %输入已知矩阵 A
A =
     1     2     3     4
     2     3     5     8
     1     3     5     7
     3     4     7    11
≫det(A)                                   %检查行列式的值是否为零，否则逆矩阵将不存
                                            在
ans =
   1
≫inv(A)                                   %矩阵 A 的逆矩阵，它应该满足inv (A)* A = eye
                                            (4) ，inv 为逆矩阵函数
ans =
    1.0000   -1.0000   -1.0000    1.0000
    2.0000    4.0000   -1.0000   -3.0000
         0   -5.0000    1.0000    3.0000
   -1.0000    2.0000   -0.0000   -1.0000
≫inv(A)* A                                %核对逆矩阵的正确性，inv (A)* A 乘积结果为
                                            单位矩阵
ans =
    1.0000   -0.0000         0   -0.0000
    0.0000    1.0000         0         0
         0    0.0000    1.0000    0.0000
         0         0   -0.0000    1.0000
```

另一种解法是通过单位矩阵左除矩阵 A，或单位矩阵右除矩阵 A 来取得。即

```
≫A \ eye(4)                               %用左除法求逆矩阵
ans =
    1.0000   -1.0000   -1.0000    1.0000
    2.0000    4.0000   -1.0000   -3.0000
         0   -5.0000    1.0000    3.0000
   -1.0000    2.0000   -0.0000   -1.0000
≫eye(4)/A                                 %用右除法求逆矩阵
```

```
ans =
     1    -1    -1     1
     2     4    -1    -3
     0    -5     1     3
    -1     2     0    -1
```

逆矩阵也可以用 A 矩阵的 -1 次乘幂来求得

```
>>A^(-1)                                %用矩阵 A 的（-1）次乘幂求逆矩阵
ans =
    1.0000   -1.0000   -1.0000    1.0000
    2.0000    4.0000   -1.0000   -3.0000
         0   -5.0000    1.0000    3.0000
   -1.0000    2.0000   -0.0000   -1.0000
```

【例 2-52】 已知 $\boldsymbol{A}=\begin{bmatrix}1&2&3&4\\2&3&5&8\\1&3&5&7\\3&4&7&11\end{bmatrix}$，用减少行成梯形最简形式函数 rref 来求逆矩阵。

解：

```
>>A = [1 2 3 4;2 3 5 8;1 3 5 7;3 4 7 11]   %已知矩阵 A
A =
     1     2     3     4
     2     3     5     8
     1     3     5     7
     3     4     7    11
>>B = [A,eye(size(A))]                  %构造增广矩阵 B，增广矩阵由矩阵 A 和同阶单
                                          位矩阵水平连结而成
B =
     1     2     3     4     1     0     0     0
     2     3     5     8     0     1     0     0
     1     3     5     7     0     0     1     0
     3     4     7    11     0     0     0     1
>>rref(B)                               %用 rref 函数使 A 矩阵成为梯形最简形式，而原
                                          来的 I 矩阵的位置成为逆矩阵
ans =
     1     0     0     0     1    -1    -1     1
     0     1     0     0     2     4    -1    -3
     0     0     1     0     0    -5     1     3
     0     0     0     1    -1     2     0    -1
```

```
≫C = ans(:,5:8)                    %取出rref(B)的5到8列即为所求逆矩阵C
C =

     1    -1    -1     1
     2     4    -1    -3
     0    -5     1     3
    -1     2     0    -1

≫inv(A)                            %核对与逆矩阵的一致性
ans =

    1.0000   -1.0000   -1.0000    1.0000
    2.0000    4.0000   -1.0000   -3.0000
         0   -5.0000    1.0000    3.0000
   -1.0000    2.0000   -0.0000   -1.0000
```

【例 2-53】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1&2&3&4\\2&3&5&8\\1&3&5&7\\3&4&7&11\end{bmatrix}$，用逆矩阵计算公式来求矩阵 $\boldsymbol{A}$ 的逆。

解：

```
≫A = [1 2 3 4;2 3 5 8;1 3 5 7;3 4 7 11]
A =

     1     2     3     4
     2     3     5     8
     1     3     5     7
     3     4     7    11

B = B* /det(A)                      % B为A的逆阵，B*为伴随矩阵，B*矩阵中
                                      的元素b(i,j)是A矩阵中的a(j,i)元素的代数
                                      余子式
```

计算程序如下。　　%有关 MATLAB 的编程，详见第 3 章

在命令窗口输入如下程序：

```
≫T = A;                             %将 A 暂存 T
≫for i = 1:4                        %设置i循环，有关循环语句for/end，详见第
                                      3章
    for j = 1:4                     %设置j循环
        A(i,:) = [];                %设置i行为空行
        A(:,j) = [];                %设置j列为空列
        b(j,i) = (-1)^(i+j)* det(A);  %计算代数余子式，并赋值给b(j,i)
        A = T;                      %将 T 复归给 A
    (end)                           %j循环结束
 end,B = b/det(A)                   %i循环结束，并显示逆矩阵B
```

```
B =                                     %计算结果

     1    -1    -1     1
     2     4    -1    -3
     0    -5     1     3
    -1     2     0    -1
```

2.35 矩阵的 LU 分解

将矩阵 $\boldsymbol{A}$ 分解为 $\boldsymbol{l}^* \boldsymbol{u}$，其中 $\boldsymbol{u}$ 为上三角矩阵，矩阵 $\boldsymbol{l}$ 为下三角矩阵。LU 分解常用于求行列式以及解线性方程组。

【例 2-54】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1&3&5&7\\2&4&6&8\\2&3&5&8\\3&4&7&11\end{bmatrix}$，求其 LU 分解。

解：在命令窗口输入如下程序：

```
>>A = [1 3 5 7;2 4 6 8;2 3 5 8;3 4 7 11]      %输入已知矩阵 A
A =

     1     3     5     7
     2     4     6     8
     2     3     5     8
     3     4     7    11

>>[l,u] = lu(A)                          %矩阵 A 的 LU 分解，lu 为 LU 分解函数
    l =                                  %程序响应，l 为心理下三角矩阵

        0.3333    1.0000         0         0
        0.6667    0.8000    1.0000         0
        0.6667    0.2000    0.2500    1.0000
        1.0000         0         0         0

    u =                                  %u 为上三角矩阵

        3.0000    4.0000    7.0000   11.0000
             0    1.6667    2.6667    3.3333
             0         0   -0.8000   -2.0000
             0         0         0    0.5000

>>l* u                                   %核对分解的正确性
  ans =

     1     3     5     7
     2     4     6     8
     2     3     5     8
     3     4     7    11
```

2.36 矩阵的正交分解

矩阵的正交分解，即矩阵的 qr 分解，正交分解是将矩阵分解成一个正交矩阵 $\boldsymbol{q}$ 和一个上三角矩阵 $\boldsymbol{r}$ 的乘积。所谓正交矩阵是该矩阵和它的转置矩阵的乘积是单位矩阵。

【例 2-55】 已知 3 阶魔方矩阵，求它的正交分解。

解：

```
≫A = magic(3)                          %A 为 3 阶魔方矩阵
A =

       8         1         6
       3         5         7
       4         9         2

≫[q,r] = qr(A)                        %对矩阵 A 进行正交分解
q =                                    %与 A 矩阵同阶的正交矩阵 q

   -0.8480    0.5223    0.0901
   -0.3180   -0.3655   -0.8748
   -0.4240   -0.7705    0.4760

r =                                    %上三角矩阵 r

   -9.4340   -6.2540   -8.1620
         0   -8.2394   -0.9655
         0         0   -4.6314

≫q* r                                  %核对分解的正确性
ans =

    8.0000    1.0000    6.0000
    3.0000    5.0000    7.0000
    4.0000    9.0000    2.0000

≫q * q'                                %检验矩阵 q 的正交性
ans =

    1.0000   -0.0000   -0.0000
   -0.0000    1.0000    0.0000
   -0.0000    0.0000    1.0000
```

2.37 矩阵的 Cholesky 分解

矩阵的 Cholesky 分解用来分解正定矩阵，它将正定矩阵分解成一个上三角矩阵 $\boldsymbol{T}$ 和 $\boldsymbol{T}$ 的转置矩阵的乘积。所谓正定矩阵，他是一个对称矩阵（与对角线对称），且其特征值全为正的矩阵。

【例 2-56】 将正定矩阵 $\boldsymbol{A}=\begin{bmatrix}1&1&1&1\\1&3&3&3\\1&3&5&5\\1&3&5&7\end{bmatrix}$进行 Cholesky 分解。

解：

```
>>A = [1 1 1 1;1 3 3 3;1 3 5 5;1 3 5 7]     %输入对称矩阵 A
A =

     1     1     1     1
     1     3     3     3
     1     3     5     5
     1     3     5     7

>>eig(A)                                    %检测 A 矩阵的特征值是否为正，以确定它的正
                                              定性。eig 为矩阵特征值的函数
ans =                                       %矩阵 A 的特征值全为正

    0.5198
    0.7232
    1.6199
   13.1371

>>chol(A)                                   %对 A 矩阵进行 Cholesky 分解
ans =

    1.0000    1.0000    1.0000    1.0000
         0    1.4142    1.4142    1.4142
         0         0    1.4142    1.4142
         0         0         0    1.4142

>>T = ans                                   %将结果赋予 T
T =

    1.0000    1.0000    1.0000    1.0000
         0    1.4142    1.4142    1.4142
         0         0    1.4142    1.4142
         0         0         0    1.4142

>>T'* T                                     %核对分解的正确性
ans =

    1.0000    1.0000    1.0000    1.0000
    1.0000    3.0000    3.0000    3.0000
    1.0000    3.0000    5.0000    5.0000
    1.0000    3.0000    5.0000    7.0000
```

2.38 广义逆矩阵

广义逆矩阵又称伪逆矩阵。当矩阵 $\boldsymbol{A}$ 的行数 m 与列数 n 不等或 $\det|\boldsymbol{A}|=0$ 时，则不存在逆矩阵。但存在广义逆矩阵 $\boldsymbol{P}$，它满足 $\boldsymbol{APA}=\boldsymbol{A}$，$\boldsymbol{PAP}=\boldsymbol{P}$，广义逆矩阵的函数为 pinv。广义逆矩阵用于求解超定方程组（线性方程组的行数大于列数的情况，即 $m>n$）的近似解，但它满足最小二乘解。广义逆矩阵也可用于求解欠定方程组（线性方程组的行数小于列数的情况，即 $m<n$）的一组特解。

【例 2-57】 已知 3 阶魔方阵，求广义逆矩阵。

解：

在命令窗口输入如下程序：

```
>>A = magic(3)                          %已知 3 阶魔方矩阵
A =

        8           1           6
        3           5           7
        4           9           2

>>rank(A)                               %A 矩阵的秩为 3，为满秩矩阵
ans =
     3
>>det(A)                                %A 矩阵的行列式不等于零
ans =
    -360
>>pinv(A)                               %A 矩阵的广义逆矩阵
ans =

     0.1472     -0.1444      0.0639
    -0.0611      0.0222      0.1056
    -0.0194      0.1889     -0.1028

>>inv(A)                                %A 矩阵的逆矩阵与广义逆矩阵相同
ans =

     0.1472     -0.1444      0.0639
    -0.0611      0.0222      0.1056
    -0.0194      0.1889     -0.1028
```

【例 2-58】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1&2&3\\4&5&6\\1&3&5\\2&5&8\\4&7&9\end{bmatrix}$，求广义逆矩阵 $\boldsymbol{P}$。

解：

在命令窗口输入如下程序：

```
>>A = [1 2 3;4 5 6;1 3 5;2 5 8;3 7 9]      %已知矩阵 A，行数大于列数
A =
     1     2     3
     4     5     6
     1     3     5
     2     5     8
     3     7     9
>>rank(A)                                  %矩阵 A 的秩为 3，因此矩阵 A 为列满秩
ans =
     3
>>pinv(A)                                  %用 pinv 计算广义逆矩阵
ans =
    0.0870    0.5435    0.0217    0.1087   -0.5000
   -0.2050   -0.3168   -0.3727   -0.5776    1.0000
    0.1304    0.0652    0.2826    0.4130   -0.5000
```

【例 2-59】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}8 & 1 & 6 & 1 & 2\\ 3 & 5 & 7 & 2 & 5\\ 4 & 9 & 2 & 3 & 8\end{bmatrix}$，求广义逆矩阵。

解：

在命令窗口输入如下程序：

```
>>A = [8 1 6 1 2;3 5 7 2 5;4 9 2 3 8]      %已知矩阵 A，行数小于列数
A =
     8     1     6     1     2
     3     5     7     2     5
     4     9     2     3     8
>>rank(A)                                  %矩阵 A 的秩为 3，因此矩阵 A 为行满秩
ans =
     3
>>pinv(A)                                  %用 pinv 计算广义逆矩阵
ans =
    0.1494   -0.1457    0.0592
   -0.0405    0.0105    0.0614
   -0.0173    0.1876   -0.1075
   -0.0055    0.0052    0.0160
   -0.0227    0.0122    0.0472
```

【例 2-60】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1&2&3\\4&5&6\\1&2&3\\1&2&3\end{bmatrix}$，求广义逆矩阵 $\boldsymbol{P}$。

解：

在命令窗口输入如下程序：

```
>>A = [1 2 3;4 5 6;1 2 3;1 2 3]     %已知矩阵 A
A =

     1     2     3
     4     5     6
     1     2     3
     1     2     3

>>rank(A)                            %矩阵 A 的秩等于 2，小于行数和列数
ans =
     2
>>pinv(A)                            %用 pinv 计算广义逆矩阵
ans =

   -0.3148    0.4444   -0.3148   -0.3148
   -0.0370    0.1111   -0.0370   -0.0370
    0.2407   -0.2222    0.2407    0.2407
```

2.39 数组与矩阵的乘幂

设 $\boldsymbol{x}$、$\boldsymbol{y}$ 为矩阵，但其行、列数必须相同，则它的数组乘幂为

V = power(x,y)

或者用

V = x.^y

表示，式中power为数组乘幂函数。若 x，y 中有一个是标量，则该式也成立。当 y = 1/2 的特殊情况，可以用V = sqrt(x)来计算。现举例如下：

【例 2-61】 设 $\boldsymbol{x}$ 为 3 阶魔方阵，$\boldsymbol{y}=\begin{bmatrix}1&2&3\\4&5&6\\7&8&9\end{bmatrix}$，求 $\boldsymbol{x}$ 的 $\boldsymbol{y}$ 次的数组乘幂。

解：

```
>>x = magic(3)                       %已知矩阵 x
x =

     8     1     6
     3     5     7
     4     9     2

>>y = [1 2 3;4 5 6;7 8 9]            %已知指数矩阵 y
y =
```

```
     1       2       3
     4       5       6
     7       8       9
>>V = power(x,y)        %数组 x 的 y 次乘幂，用乘幂函数 power
V =

          8            1          216
         81         3125       117649
      16384     43046721          512
>>V = x.^y              %用指数乘幂形式，两者结果是相同的
V =

          8            1          216
         81         3125       117649
      16384     43046721          512
```

【例 2-62】 已知 $x=9$，$\boldsymbol{y}$ 为 3 阶魔方阵，求 x 的 $\boldsymbol{y}$ 次的数组乘幂。

解：

```
>>x = 9;
>>y = magic(3);
>>V1 = x.^y
V1 =

   43046721            9       531441
        729        59049      4782969
       6561    387420489           81
```

对于矩阵 $\boldsymbol{A}$ 的乘幂，它的指数必须是标量，否则显示出错。矩阵的乘幂函数为 mpower，power 前的 m 表示矩阵（matrix）。它的书写格式为

G = mpower(A,y)

或者可采用

G = A^y

形式表示。现举例如下：

【例 2-63】 已知矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}$，求 $\boldsymbol{A}$ 的数组和矩阵的 1/2 次乘幂。

解：

```
>>A = [1 2;3 4]         %已知矩阵 A
A =
     1      2
     3      4
>>sqrt(A)               %数组 A 的 1/2 乘幂，相当于矩阵 A 每一个元素的开平方根
ans =

    1.0000    1.4142
```

```
    1.7321        2.0000
>>G = mpower(A,1/2)     %矩阵 A 的 1/2 次乘幂
G =

    0.5537 + 0.4644i    0.8070 - 0.2124i
    1.2104 - 0.3186i    1.7641 + 0.1458i
>>G1 = A^(1/2)          %用常用形式表示，其结果是一致的
G =

    0.5537 + 0.4644i    0.8070 - 0.2124i
    1.2104 - 0.3186i    1.7641 + 0.1458i
>>G*G                   %核对计算的正确性
ans =

    1.0000              2.0000 - 0.0000i
    3.0000 - 0.0000i    4.0000
```

2.40 矩阵的水平连接和垂直连接

若矩阵 ***A*** 和***B***，需要进行水平连接，则水平连结函数为horzcat(A,B)，但矩阵 ***A***、***B*** 的行数必须相等，否则将不能进行水平连接并显示出错。水平连接还可以用[A,B]来实现。

与水平连接相类似的是矩阵的垂直连接，矩阵垂直连接的函数为 vertcat（A，B），垂直连接时，矩阵 A，B 的列数必须相等，否则将不能进行垂直连接并显示出错。垂直连接还可以用［A；B］来表示。下面分别举例予以说明。

【例 2-64】 已知矩阵 ***A*** 为 4 阶魔方阵，矩阵 $\boldsymbol{B}=\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\\1&1&1\end{bmatrix}$，求这两个矩阵的水平连接矩阵。

解：在命令窗口键入如下程序：

```
>>A = magic(4)          %已知矩阵 A
A =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
>>B = ones(4,3)         %已知矩阵 B
B =

     1     1     1
     1     1     1
     1     1     1
```

```
     1     1     1
≫C = horzcat(A,B)           %矩阵 A、B 经水平连接后的矩阵 C
C =

    16     2     3    13     1     1     1
     5    11    10     8     1     1     1
     9     7     6    12     1     1     1
     4    14    15     1     1     1     1
```

【例 2-65】 已知矩阵 A 为 2 行 5 列随机矩阵，矩阵 $B=\begin{bmatrix}1 & 2 & 3 & 4 & 5\\3 & 5 & 7 & 9 & 11\end{bmatrix}$，求这两个矩阵的垂直连接矩阵。

解：

在命令窗口键入如下程序：

```
≫A = rand(2,5)              %已知均匀分布的 2 行 5 列的随机矩阵 A
A =

    0.9501    0.6068    0.8913    0.4565    0.8214
    0.2311    0.4860    0.7621    0.0185    0.4447
≫B = [1 2 3 4 5;3 5 7 9 11]  %已知 2 行 5 列的整数矩阵 B
B =

     1     2     3     4     5
     3     5     7     9    11
≫C = vertcat(A,B)           %矩阵 A、B 经垂直连接后的矩阵 C
C =

    0.9501    0.6068    0.8913    0.4565    0.8214
    0.2311    0.4860    0.7621    0.0185    0.4447
    1.0000    2.0000    3.0000    4.0000    5.0000
    3.0000    5.0000    7.0000    9.0000   11.0000
```

2.41 矩阵的复制

当矩阵的阶次在 4 阶以下，矩阵元素的输入可以逐个写入。但是当矩阵阶次较大时，且结构相同时，用逐个输入的方法太费时间，这时可以用矩阵复制的办法来加快输入的速度。矩阵的复制函数为

repmat(A,m,n)

式中，A 为待复制的向量或矩阵，m、n 为需要复制的行数和列数。

【例 2-66】 已知向量 $V=[2\ 3\ 5\ 8]$，求矩阵 B，它复制 10 行向量 V。

解：

在命令窗口输入如下程序：

```
≫V = [2 3 5 8];             %已知向量 V
```

```
≫m = 10; n = 1;                %复制行数和列数
≫B = repmat(V, m, n)           %复制向量
B =                            %复制完成后的矩阵 B

     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
     2     3     5     8
```

【例 2-67】 已知矩阵 $\boldsymbol{A}$ 为 3 阶 Pascal 矩阵，求矩阵 $\boldsymbol{B}_1$，它复制 3 行 4 列矩阵 $\boldsymbol{A}$。

解：

在命令窗口输入如下程序：

```
≫A = pascal(3)
A =

     1     1     1
     1     2     3
     1     3     6

≫B1 = repmat(A, 3, 4)
B1 =

     1     1     1     1     1     1     1     1     1     1     1     1
     1     2     3     1     2     3     1     2     3     1     2     3
     1     3     6     1     3     6     1     3     6     1     3     6
     1     1     1     1     1     1     1     1     1     1     1     1
     1     2     3     1     2     3     1     2     3     1     2     3
     1     3     6     1     3     6     1     3     6     1     3     6
     1     1     1     1     1     1     1     1     1     1     1     1
     1     2     3     1     2     3     1     2     3     1     2     3
     1     3     6     1     3     6     1     3     6     1     3     6
```

【例 2-68】 已知矩阵 $\boldsymbol{A}_1$ 为 3 阶魔方阵，将 $\boldsymbol{A}_1$ 复制到 9×9 矩阵 $\boldsymbol{B}_2$ 的主对角线上，而矩阵 $\boldsymbol{B}_2$ 的其他元素均为零。

解：

在命令窗口输入如下程序：

```
≫A1 = magic(3)                 %已知矩阵 A1
A1 =
```

```
     8     1     6
     3     5     7
     4     9     2
≫Z = zeros(3)                     %设置3阶零矩阵
Z =

     0     0     0
     0     0     0
     0     0     0
≫B2 = [A1 Z Z;Z A1 Z;Z Z A1]      %在矩阵B2对角线上复制矩阵A1
B2 =

     8     1     6     0     0     0     0     0     0
     3     5     7     0     0     0     0     0     0
     4     9     2     0     0     0     0     0     0
     0     0     0     8     1     6     0     0     0
     0     0     0     3     5     7     0     0     0
     0     0     0     4     9     2     0     0     0
     0     0     0     0     0     0     8     1     6
     0     0     0     0     0     0     3     5     7
     0     0     0     0     0     0     4     9     2
```

2.42 稀疏矩阵的创建

当矩阵的大部分元素是零，只有少数元素为非零元素时，这种矩阵称为稀疏矩阵。

在【例2-68】中的矩阵 $\boldsymbol{B}_2$，就是稀疏矩阵的例子。用矩阵来存储稀疏矩阵，要占用较大的存储空间，因此要用稀疏矩阵表示法来节省存储空间。稀疏矩阵可以用以下几种方法来建立。

(1) 用稀疏矩阵函数 sparse（A）直接建立，其中A为原来的全矩阵。

【例2-69】 已知6阶单位矩阵 $\boldsymbol{A}$，求其稀疏矩阵 $\boldsymbol{B}$。

解：

在命令窗口输入如下程序：

```
≫A = eye(6)                       %输入6阶单位矩阵
A =

     1     0     0     0     0     0
     0     1     0     0     0     0
     0     0     1     0     0     0
     0     0     0     1     0     0
     0     0     0     0     1     0
     0     0     0     0     0     1
≫B = sparse(A)                    %转换成稀疏矩阵
B =
```

```
    (1,1)        1
    (2,2)        1
    (3,3)        1
    (4,4)        1
    (5,5)        1
    (6,6)        1
≫whos                        %显示矩阵 A，B 在工作空间所占 byte 数，显然，矩阵
                               B 所占空间远小于矩阵 A
  Name        Size        Bytes  Class
  A           6x6           288  double array
  B           6x6           100  sparse array
Grand total is 42 elements using 388 bytes
```

（2）用非零元素直接输入创建稀疏矩阵。

用非零元素直接输入，创建稀疏矩阵的格式为

S = sparse(i,j,s,m,n)

式中 i、j 为非零元素所在的行和列的整数向量；S 为对应 i、j 向量的元素值的向量；m、n 为稀疏矩阵总的行数和列数。

【例 2-70】 已知 $\boldsymbol{i}$ = [1 1 1 2 2 2 3 3 3 4 4 4]，$\boldsymbol{j}$ = [1 3 5 2 4 6 3 5 7 4 6 8]，$\boldsymbol{s}$ = [2 5 13 3 8 21 5 13 34]，$m=6$，$n=8$，求稀疏矩阵 $\boldsymbol{S}$。

解：

在命令窗口输入如下程序：

```
≫i = [1 1 1 2 2 2 3 3 3 4 4 4];
≫j = [1 3 5 2 4 6 3 5 7 4 6 8];
≫s = [2 5 13 3 8 21 5 13 34 8 21 55];
≫m = 6; n = 8;
≫S = sparse(i,j,s,m,n)               %设定参数的稀疏矩阵
S =
    (1,1)        2
    (2,2)        3
    (1,3)        5
    (3,3)        5
    (2,4)        8
    (4,4)        8
    (1,5)       13
    (3,5)       13
    (2,6)       21
    (4,6)       21
    (3,7)       34
```

```
    (4,8)        55
>>full(S)                    %稀疏矩阵转换成全矩阵。full为转换函数
ans =
     2     0     5     0    13     0     0     0
     0     3     0     8     0    21     0     0
     0     0     5     0    13     0    34     0
     0     0     0     8     0    21     0    55
     0     0     0     0     0     0     0     0
     0     0     0     0     0     0     0     0
>>nnz(S)                     %稀疏矩阵S中含有非零元素数，nnz为非零元素个数的计数
                              函数
ans =
    12
>>density = ans/(m*n)        %稀疏矩阵S的密度定义为非零元素数与矩阵总元素之比。计
                              算结果为25%
density =
     0.2500
```

（3）从对角线方式产生稀疏矩阵。

从对角线方式产生稀疏矩阵的格式为

A = spdiags(B,d,m,n)

它是将矩阵B的每一列，按整数向量d的对应元素的值，写入矩阵A的对角线上，如果d的某一列元素为零，则写入矩阵A的主对角线上，若d的某一列为正整数，则相应上移整数行。若d的某一列为负整数，则相应下移整数行。若填写的对角线元素超出矩阵A的对角线范围，则删除。注意矩阵B的列数必需与向量d的列数相等，并且矩阵B的行数必需等于min(m,n)(即m，n中最小的一个，m，n为矩阵A的行数和列数）否则将显示出错。

下面举例予以说明。

【例2-71】 已知由随机矩阵产生的4行5列的0、1矩阵$\boldsymbol{A}$，设置向量$\boldsymbol{d}=[-2\ \ -1\ 0\ 1\ 2]$，$m=4$，$n=6$，试用对角线方式产生的稀疏矩阵$\boldsymbol{B}$。

解：

```
>>A = 2*rand(4,5)            %设置均匀分布的随机矩阵，并乘以2
A =
    1.1871    1.2898    0.5795    0.6186    1.4055
    0.9931    1.6359    0.6824    1.6770    1.0931
    1.7995    1.3205    1.0682    1.1361    0.8898
    1.6433    0.6839    1.4542    0.7408    1.3891
>>A = A > 1                  %取矩阵A元素，若大于1的置1，否则取零，则成0，1分
                              布的矩阵，并赋值给新的矩阵A
A =
```

```
    1    1    0    0    1
    0    1    0    1    1
    1    1    1    1    0
    1    0    1    0    1
≫B = spdiags(A,[-2,-1,0,1,2],4,6)     % 用对角线方式产生的稀疏矩阵，这里size(A,2)
                                       = size(d),size(A,1) = 4
B =
    (2,1)       1
    (3,1)       1
    (3,2)       1
    (4,2)       1
    (1,3)       1
    (2,3)       1
    (3,3)       1
    (2,4)       1
    (3,4)       1
    (4,4)       1
    (4,6)       1
≫C = full(B)                          % 相应的全矩阵
C =
    0    0    1    0    0    0
    1    0    1    1    0    0
    1    1    1    1    0    0
    0    1    0    1    0    1
```

【例 2-72】 已知矩阵 $\boldsymbol{A}$ = repmat([1 2 1],5,1),$\boldsymbol{d}$ = [-1 0 1]，$m = n = 5$，求对角稀疏矩阵 $\boldsymbol{B}$ = spdiags(A,d,m,n)。

解:

```
≫A = [1 2 1];                         % 设置向量 A
≫A = repmat(A,5,1)                    % 复制向量 A 成 5 行
A =
    1    2    1
    1    2    1
    1    2    1
    1    2    1
    1    2    1
≫B = spdiags(A,-1:1,5,5)              % 用对角线方式产生的稀疏矩阵，这里d = [-1,
                                       0,1],m = n = 5
```

```
B =
    (1,1)        2
    (2,1)        1
    (1,2)        1
    (2,2)        2
    (3,2)        1
    (2,3)        1
    (3,3)        2
    (4,3)        1
    (3,4)        1
    (4,4)        2
    (5,4)        1
    (4,5)        1
    (5,5)        2
>>C = full(B)                          %转换成全矩阵
C =
    2     1     0     0     0
    1     2     1     0     0
    0     1     2     1     0
    0     0     1     2     1
    0     0     0     1     2
```

2.43 稀疏矩阵的图形显示

为了形象地显示稀疏矩阵的稀疏密度，可以用稀疏矩阵图像化函数spy，其语法格式如下：

spy(S)

spy(S，markersize)

spy(S，'LineSpec'，marksize)

式中，S为稀疏矩阵，markersize为标记尺寸，用整数表示。LineSpec指线型的规范，在这里指标记的类型。有关LineSpec的详细情况将在"第5章数据的可视化"中叙述。

格式spy(S)，输出任意稀疏矩阵的图形，其横坐标表示列数，纵坐标表示行数。该图形中对非零元素相应的坐标显示一个圆点，圆点的颜色为蓝色。这圆点的大小与非零元素的数值无关。对于稀疏矩阵中的零元素相应位置，则显示空白。

对于格式spy(S，markersize)的输出与spy(S)相似，所不同的是前者可以通过markersize来改变圆点的大小。

对于格式spy(S，'LineSpec'，markersize)，则可以通过LineSpec来改变标记的类型和颜色。

下面举例说明稀疏矩阵的图形显示。

【例2-73】 显示稀疏对角矩阵B = spdiags(A，[-2，-1，0，1，2]，8，12)，其中A = magic

(8) > 30，即在 8 阶魔方矩阵中元素大于 30 的置 1，等于或小于 30 的置零，随后赋予矩阵 A。

解：

在命令窗口输入如下程序：

```
>>A = magic(8) > 30                  %检测 8 阶魔方矩阵元素大于 30 的返回
                                       1，否则返回零，并赋入矩阵 A
  A =
     1    0    0    1    1    0    0    1
     0    1    1    0    0    1    1    0
     0    1    1    0    0    1    1    0
     1    0    0    1    1    0    1    1
     1    1    1    0    0    1    1    0
     1    0    0    1    1    0    0    1
     1    0    0    1    1    0    0    1
     0    1    1    0    0    1    1    0
>>B = spdiags(A,[-2,-1,0,1,2],8,12)   %建立稀疏对角矩阵 B
B =
     (2,1)      1
     (1,2)      1
     (2,2)      1
     (3,2)      1
     (4,2)      1
     (1,3)      1
     (3,3)      1
     (5,3)      1
     (5,4)      1
     (6,4)      1
     (4,5)      1
     (5,5)      1
     (7,5)      1
     (4,6)      1
     (6,7)      1
     (8,7)      1
     (6,8)      1
     (7,8)      1
     (8,8)      1
     (7,9)      1
>>spy(B)                              %显示稀疏矩阵 B 的图形如图 2-2 所示
```

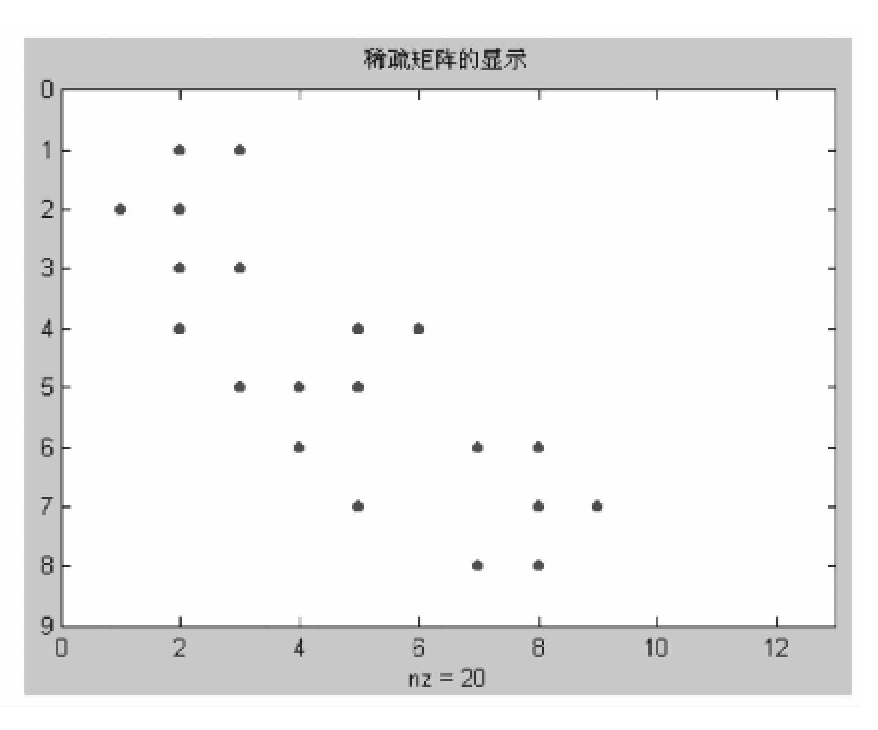


图 2-2　稀疏对角矩阵的显示

【例 2-74】　同上例，但改变标记为星形，标记尺寸改为 15。

解:

在命令窗口输入如下程序：

```
>>A = magic(8) > 30;                          %判别8阶魔方矩阵元素大于30的返回1,否则
                                                返回零，并赋入矩阵 A
>>B = spdiags(A,[-2，-1,0,1,2],8,12);         %建立稀疏对角矩阵 B
>>spy(B,'*',15)                               %显示稀疏矩阵 B，用“*”号标记，标记尺寸为
                                                15，如图 2-3 所示
```

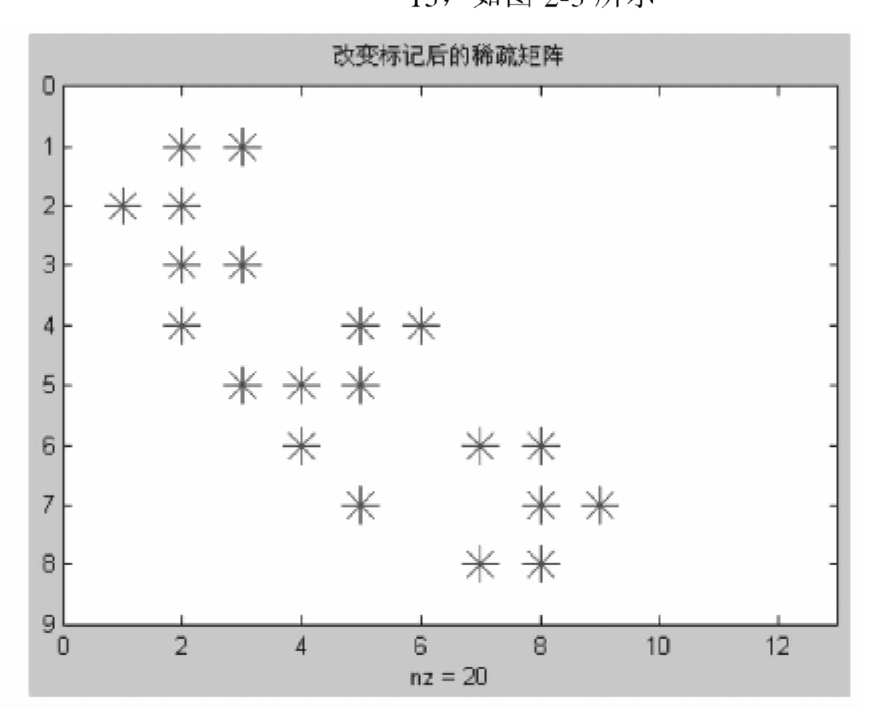


图 2-3　改变标记为星形，尺寸改为 15 后的稀疏矩阵

2.44 寻找矩阵的非零元素

为了寻找稀疏矩阵的非零元素所在位置，函数 find 也是十分有用的。find 函数的书写格式为

k = find(X)

[i,j] = find(X)

[i,j.v] = find(X)

式中，X 为矩阵，k 为矩阵 X 中非零元素的一维下标索引，i，j 为矩阵 X 中的非零元素的所在行和列的下标，v 为非零元素所在行、列的元素值。现举例于以说明。

【例 2-75】 已知稀疏矩阵 ***B***，其主对角线元素为 ***A*** = [2 3 5 8 13 21 34]，主对角线上一行和下一行元素均为 1，其他元素则全为零。求稀疏矩阵 ***B*** 的非零元素的一维索引 k，二维索引 i，j 及其相应元素值 v。

解：

在命令窗口输入如下程序：

```
≫A = [2 3 5 8 13 21 34];                 %已知向量 A
≫E = ones(6,1);                          %主对角线上一行和下一行向量
≫B = diag(A) + diag(E,1) + diag(E,-1)    %建立稀疏矩阵 B
B =

     2     1     0     0     0     0     0
     1     3     1     0     0     0     0
     0     1     5     1     0     0     0
     0     0     1     8     1     0     0
     0     0     0     1    13     1     0
     0     0     0     0     1    21     1
     0     0     0     0     0     1    34

≫k = find(B);                            %稀疏矩阵 B 的一维索引
≫k'                                      %以行向量显示
ans =
   Columns 1 through 14
     1     2     8     9    10    16    17    18    24    25    26    32    33    34
   Columns 15 through 19
    40    41    42    48    49
≫[i,j] = find(B);                        %稀疏矩阵的二维索引 i，j
≫i'                                      %i 的行向量显示
ans =
   Columns 1 through 14
```

```
   1     2     1     2     3     2     3     4     3     4     5     4     5     6
  Columns 15 through 19
   5     6     7     6     7
>>j'                          %j的行向显示
ans =
  Columns 1 through 14
   1     1     2     2     2     3     3     3     4     4     4     5     5     5
  Columns 15 through 19
   6     6     6     7     7
>>[i,j,v] = find(B);          %稀疏矩阵B的二维索引和相应的元素值
>>v'                          %以行向量显示
ans =
  Columns 1 through 14
   2     1     1     3     1     1     5     1     1     8     1     1    13     1
  Columns 15 through 19
   1    21     1     1    34
>>nnz(B)                      %显示非零元素的个数
ans =
    19
```

第 2 章习题

2-1 求 20 ~ 100 的线性向量间隔为 4。

2-2 求 1 ~ 1000 间分 10 个点的对数分布向量。

2-3 求 sin10°，sin20°，…，sin90°的值，并求其代数和。

2-4 已知 x 为 1 ~ 10 的整数，求 $y = (x + x^2 + x^3)/(1 + \sin x/2)$。

2-5 输入下列矩阵，并求其行列式值。

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

2-6 已知矩阵 $\boldsymbol{A} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 2 & 1 \\ 3 & 1 & 1 & 3 \end{bmatrix}$，矩阵 $\boldsymbol{B} = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \\ 1 & 3 & 5 \\ 5 & 3 & 1 \end{bmatrix}$，计算矩阵 $\boldsymbol{A}$，$\boldsymbol{B}$ 的矩阵乘积。

2-7 已知矩阵 $\boldsymbol{A}$ 为 4 阶魔方阵，矩阵 $\boldsymbol{B} = \begin{bmatrix} 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \\ 5 & 6 & 7 & 8 \end{bmatrix}$，求矩阵 $\boldsymbol{A}$、$\boldsymbol{B}$ 的数组乘积。

2-8　已知矩阵 $A=\begin{bmatrix}1&2&3&4&5\\1&3&6&10&15\\1&4&10&20&25\end{bmatrix}$，$B=\begin{bmatrix}17&24&15\\23&5&16\\4&6&22\\10&12&3\\11&18&9\end{bmatrix}$，求矩阵 A、B 的积 C。矩阵 C 是方阵吗？若是，则求其行列式值。

2-9　已知矩阵 $A=\begin{bmatrix}64&118&155\\121&220&290\\178&322&425\end{bmatrix}$，$B=\begin{bmatrix}3&4&5\\6&10&15\\10&20&25\end{bmatrix}$，求 A/B。

2-10　已知矩阵 $A=\begin{bmatrix}1&1&2\\1&3&4\\2&4&5\end{bmatrix}$，矩阵 $B=\begin{bmatrix}14&18\\30&36\\39&50\end{bmatrix}$，求 $A\ \backslash B$。

2-11　计算下列行列式：

(1) $\begin{vmatrix}2&0&17&1\\1&5&3&3\\3&1&-1&2\\-5&4&8&11\end{vmatrix}$　　(2) $\begin{vmatrix}1&3&5&7&9\\2&4&6&8&10\\5&6&7&6&5\\10&8&6&4&2\\9&7&5&3&1\end{vmatrix}$

2-12　已知矩阵 $A=\begin{vmatrix}1&1&1&1\\8&4&2&1\\27&9&3&1\\64&16&4&1\end{vmatrix}$，求转置矩阵。

2-13　用 size 命令测试下列矩阵的大小。

(1) $\begin{bmatrix}61&92&17&93&41\\79&73&40&91&89\end{bmatrix}$　　(2) $\begin{bmatrix}-43&28&118&17\\-166&-114&-3&-18\\12&119&32&72\end{bmatrix}$

2-14　求下列矩阵是否满秩。

(1) $\begin{bmatrix}16&2&3&13\\5&11&10&8\\9&7&6&12\\4&14&15&1\end{bmatrix}$　　(2) $\begin{bmatrix}100&50&33&25&20\\50&33&25&20&16\\33&25&20&16&14\\25&20&16&14&12\\20&16&14&12&11\end{bmatrix}$

2-15　求下列向量的 2 范数。

(1) $V=[1\ 3\ 5\ 7]$

(2) $V_1=[1\ -2\ 5\ 8]$

2-16　求下列矩阵的 2 范数。

(1) $\begin{bmatrix}3&3&6&2&4\\3&9&12&6&8\\6&12&15&8&10\end{bmatrix}$　　(2) $\begin{bmatrix}1.0000&0.5000&0.3333&0.2500\\0.5000&0.3333&0.2500&0.2000\\0.3333&0.2500&0.2000&0.1667\\0.2500&0.2000&0.1667&0.1429\end{bmatrix}$

2-17　求下列矩阵的条件数。

(1) $\begin{bmatrix}5&4&0&0\\1&5&4&0\\0&1&5&4\\0&0&1&5\end{bmatrix}$　　(2) $\begin{bmatrix}0&4&4&1\\2&3&3&0\\3&2&0&3\\4&0&1&4\end{bmatrix}$

2-18　计算下列矩阵的秩。

(1) $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 9 & 11 \\ 10 & 13 & 16 \\ 14 & 17 & 20 \end{bmatrix}$　　(2) $\begin{bmatrix} 16 & 2 & 3 & 13 \\ 5 & 11 & 10 & 8 \\ 9 & 7 & 6 & 12 \\ 4 & 14 & 15 & 1 \end{bmatrix}$

2-19　计算下列矩阵的特征值和特征向量。

(1) $\begin{bmatrix} 8 & 1 & 6 \\ 3 & 5 & 7 \\ 4 & 9 & 2 \end{bmatrix}$　　(2) $\begin{bmatrix} -10 & -35 & -50 & -24 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

2-20　用函数 ones 和 diag 分别编写下列矩阵。

(1) $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 3 & 3 & 3 & 3 & 1 \\ 1 & 3 & 5 & 5 & 5 & 3 & 1 \\ 1 & 3 & 5 & 7 & 5 & 3 & 1 \\ 1 & 3 & 5 & 5 & 5 & 3 & 1 \\ 1 & 3 & 3 & 3 & 3 & 3 & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}$　　(2) $\begin{bmatrix} 4 & 5 & 6 & 0 & 0 \\ 3 & 4 & 5 & 6 & 0 \\ 2 & 3 & 4 & 5 & 6 \\ 0 & 2 & 3 & 4 & 5 \\ 0 & 0 & 2 & 3 & 4 \end{bmatrix}$

2-21　将矩阵 $\boldsymbol{A}=\begin{bmatrix} 16 & 4 & 8 & 4 \\ 4 & 10 & 8 & 4 \\ 8 & 8 & 12 & 10 \\ 4 & 4 & 10 & 12 \end{bmatrix}$进行 lu 分解，并计算它的行列式。

2-22　已知 $\boldsymbol{G}=\begin{vmatrix} \boldsymbol{A} & \boldsymbol{C} \\ \boldsymbol{D} & \boldsymbol{B} \end{vmatrix}$，其中 $\boldsymbol{A}$、$\boldsymbol{B}$、$\boldsymbol{C}$、$\boldsymbol{D}$ 为符号块矩阵，求 $\boldsymbol{G}$ 的逆矩阵。

2-23　求下列矩阵的逆矩阵。

(1) $\begin{bmatrix} 1 & 2 & 8 \\ 1 & 3 & 5 \\ 2 & 1 & 5 \end{bmatrix}$　　(2) $\begin{bmatrix} 5 & 8 & 13 & 0 & 0 \\ 2 & 5 & 8 & 13 & 0 \\ 1 & 2 & 5 & 8 & 13 \\ 0 & 1 & 2 & 5 & 8 \\ 0 & 0 & 1 & 2 & 5 \end{bmatrix}$

2-24　用减小行成梯形最简形式函数 rref 求下列矩阵的逆矩阵。

(1) $\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 3 & 6 & 10 \\ 1 & 4 & 10 & 20 \end{bmatrix}$　　(2) $\begin{bmatrix} 11 & 13 & 0 & 0 \\ 8 & 11 & 13 & 0 \\ 0 & 8 & 11 & 13 \\ 0 & 0 & 8 & 11 \end{bmatrix}$

2-25　求下列矩阵的广义逆矩阵。

(1) $\begin{bmatrix} 1 & 2 & 3 \\ 2 & 5 & 8 \\ 2 & 4 & 6 \\ 4 & 10 & 16 \\ 3 & 6 & 9 \end{bmatrix}$　　(2) $\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 0 & 3 & 5 & 6 & 0 \\ 0 & 0 & 5 & 8 & 9 \end{bmatrix}$

第 3 章　MATLAB 编程与数据类型

在科学和工程计算中，对于比较简单的问题，可以直接在命令窗口输入原始数据，调用相关的内装函数，如三角函数、指数函数、统计分析函数、矩阵运算函数、图形绘制函数、微分方程求解函数和最优化函数等，接着进行分析计算即可。对于比较复杂的问题或者需要重复使用的问题，则应该编写程序文本文件或函数文本文件进行程序运算。

本章介绍文本 M 文件和函数 M 文件的编制，编程用的流程控制语句，如 for/end、while/end、if/eiseif/、…、else/end、switch/case/end、try/catch/end、continue、break、return 等。在数据类型方面，则介绍数值型数组、字符型数组、单元数组和结构数据等的构成及其应用。

3.1　函数 M 文件

MATLAB 有 1000 余条内装函数，如三角函数、幂函数、指数函数、多项式函数、插值函数、求多项式根函数、矩阵运算函数、微分函数、积分函数、微分方程求解函数和图形绘制函数等。这些函数都用后缀为 M 的函数文件编写，故又称函数 M 文件。由于大量内装函数的存在，读者只要掌握函数的调用和使用规则即可实现科学计算和绘图功能，省去了大量的、繁琐的函数编程工作，使读者为解决实际问题的编程工作大为简化。当调用 MATLAB 函数时，在命令窗口输入函数名，则该函数的程序即被调用并执行。下面举例予以说明。

【例 3-1】　求解 3 次代数方程式

$$y = x^3 + 6x^2 + 11x + 6$$

的根，则可以在 MATLAB 命令窗口进行下列操作，便可完成。

解：

```
≫syms x                          %设置变量 x 为符号变量
≫y = x^3 + 6* x^2 + 11* x + 6;   %将 3 次代数方程式赋予变量 y
≫y1 = sym2poly(y)                %提取 3 次代数方程的系数向量 y1，sym2poly 为将多项
                                    式转换成系数向量的转换函数
y1 =
        1        6        11        6
≫roots(y1)                       %求解方程式的根
ans =                            %答案
    -3.0000
    -2.0000
    -1.0000
```

上述操作中，若直接输入系数向量 y1，则程序开始的 3 个语句可省略。

如果用二分法解代数多项式的 1 个实根的 QBASIC 程序，求解上述方程式的根，则程序

如下：

在 QBASIC 的工作窗口输入如下程序：

```
10 INPUT a,b,e                     %输入实根所在的区间[a,b]和允许误差 e
20 IF ABS(b-a)>=e THEN 40C         %判别实根所在的区间[a,b]是否小于允许误差，若大
                                     于则转向语句 40
30 GOTO 80                         %否则转向语句 80 即打印输出语句
40 x=(a+b)/2 :GOSUB 100            %取根所在区间的中点，转向语句 100 的计算多项式函
                                     数子程序再试算
50 IF F>=0 THEN 70                 %若多项式的值大于、等于零则转向语句 70，否则转向
                                     语句 60
60 a=x:GOTO 20                     %缩小左侧搜索区，并转向语句 20
70 B=x:GOTO 20                     %缩小右侧搜索区，并转向语句 20
80 PRINT 'x=';x                    %输出答案
90 END                             %程序结束
100 F=x^3+6*x^2+11*x+6             %函数子程序
110 RETURN                         %返回
```

运行该程序结果如下：

```
? -1.5,0,1e-5                      %输入根所在的区间和允许误差
x=-1.000002                        %用二分法解代数方程式的实根
```

当使用 QBASIC 求出 1 个实根后，尚需搜索第 2、3 个根的所在区间，所以比较费时，并且该程序尚无法求解复根。但是采用 MATLAB 求多项式根的函数 roots 时，则比较方便，当调用该函数，并输入多项式系数向量，则多项式的根，包括复根在内，全部被解出。

显然用 MATLAB 比 QBASIC 解上述问题大为方便。

3.2 函数 M 文件的组成

M 文件的编写在 M 文件编辑器中进行。点击文件菜单 file/new/m-file，即可打开 M 文件编辑器。如图 3-1 所示。

函数 M 文件是由下列 5 部份组成：

(1) 函数定义行。它必须由关键字 function 开头，紧跟着是函数的输出变量，如果有多个输出变量则需用方括号括起来，各输出变量间用逗号隔开。在等式右边则为函数名，后面紧接着是函数的输入变量，并用圆括号括起来，如果有多个输入变量，则也用逗号分隔。函数名与变量一样，必须是由字母开头的，由字母、数字和下划线组成（不能含其他符号），总长度不得超过 31 字符。

(2) 帮助文本的标题行，简称 H1 行（即 Help 的第 1 行），这一行简明扼要地说明函数的功能。

(3) 帮助文本的内容，详细说明变量的类型，使用时的语法规则，使用举例和相关的函数名作为查找时的参考。

(4) 函数体。即由输入变量计算输出变量的程序体，还包括程序运行时的出错处理。

(5) 附注。说明函数的编者、版权和日期。

```
C:\MATLAB6p5p1\work\triarea.m
File Edit View Text Debug Breakpoints Web Window Help
Stack: Base

1   function A=triarea(a,b,c)
2 - if a<0 |b<0 |c<0
3 -     disp('The a,b,c must be positive number.'),break
4 - end
5 - if a+b<c |a+c<b |b+c<a
6 -     disp('This is impossible.'),break
7 - end
8 -         s=(a+b+c)/2;
9 -         A=(s*(s-a)*(s-b)*(s-c))^(1/2);
10  % TRIAREA(a,b,c) is calculate the area of triangle.
11  % The a,b,c must a scale and nonnegtive.
12  % and any two sides of triangle must be great then the third.
13  % For example triarea(3,4,5)
14  %            ans=
15  %                 6
16  %            triarea(3,0,8)
17  %            ans=
18  %                 0
19  %            triarea(11,18,6)
20  %            this is impossible.
21  %            triarea(-5,6,7)
22  %            The a,b,c must be positive number.
23  % See also POLTAREA
24
25

triarea    Ln 1    Col 1
```

图 3-1　M 文件编辑器

今以简单的函数为例，查看计算平均值 mean 的程序是如何编著的。在命令窗口输入

```
>>type mean                                        % 列出 mean 的函数文件
```

则显示 mean 的函数文件内容如下（下面定义行和函数体的附注以及帮助行后第 2 层附注为作者所加）：

```
function y = mean(x,dim)                           % 函数定义行，mean 为函数名，x、dim 为输入变量，y 为输出变量
%   MEAN   Average or mean value.                  % H1 行，平均数或平均值
%   For vectors,MEAN(X) is the mean value of the elements in X.For    % 以下 15 行为帮助文木内容
%   matrices,MEAN(X) is a row vector containing the mean value of
%   each column.For N-D arrays,MEAN(X) is the mean value of the
%   elements along the first non-singleton dimension of X.
%
```

```
%   MEAN(X,DIM) takes the mean along the dimension DIM of X.
%
%   Example: If X = [0 1 2;3 4 5]
%
%
%   then mean(X,1) is [1.5 2.5 3.5] and mean(X,2) is [1;4]
%
%
%   See also MEDIAN,STD,MIN,MAX,COV.
%   Copyright 1984-2001 The MathWorks,Inc.        % 以下两行，说明版权、版本和编著时间
%   $Revision:5.16 $ $Date:2001/04/15 12:01:26 $
if nargin == 1,                                   %以下为 mean 函数的函数体，nargin 为输入变量数
    % Determine which dimension SUM will use
    dim = min(find(size(x) ~ = 1));
    if isempty(dim), dim = 1; end
    y = sum(x)/size(x,dim);
else
    y = sum(x,dim)/size(x,dim);
end                                               %程序结束
```

对于多输入、多输出函数的例子之一是 cylinder（产生一个多边形棱柱体），它的定义行为

function [xx,yy,zz] = cylinder(r,n)

式中，xx、yy、zz 为 3 个坐标的输出矩阵，r 为多边形棱柱体分段外接圆向量，n 为多边形边数。

其程序如下：

```
>>type cylinder
function [xx,yy,zz] = cylinder(r,n)
%   CYLINDER Generate cylinder.
%   [X,Y,Z] = CYLINDER(R,N) forms the unit cylinder based on the generator
%   curve in the vector R. Vector R contains the radius at equally
%   spaced points along the unit height of the cylinder. The cylinder
%   has N points around the circumference. SURF(X,Y,Z) displays the
%   cylinder.
%   [X,Y,Z] = CYLINDER(R), and [X,Y,Z] = CYLINDER default to N = 20
%   and R = [1 1].
```

```
%   Omitting output arguments causes the cylinder to be displayed with
%   a SURF command and no outputs to be returned.
%   See also SPHERE, ELLIPSOID.
%   Clay M. Thompson 4-24-91, CBM 8-21-92.
%   Copyright 1984-2001 The MathWorks, Inc.
%   $Revision: 5.7 $ $Date:2001/04/15 12:03:51$
if nargin < 2,n = 20;end
if nargin < 1,r = [1 1]'; end
r = r(:); % Make sure r is a vector.
m = length(r); if m = = 1,r = [r;r]; m = 2; end
theta = (0:n)/n*2*pi;
sintheta = sin(theta); sintheta(n + 1) = 0;
x = r*cos(theta);
y = r*sintheta;
z = (0:m - 1)'/(m - 1)*ones(1,n + 1);
if nargout = = 0
    surf(x,y,z)
else
    xx = x; yy = y; zz = z;
end
```

但 MATLAB 有的函数 M 文件的程序是被隐藏的，当用 type 命令显示函数 M 程序文件时，命令窗口会显示，这是内装函数，因而得不到函数 M 程序文件内容。例如要想列出求矩阵逆函数的 M 文件时，在命令窗口键入 type inv，结果显示如下：

```
>>type inv
inv is a built-in function.
```

它告诉用户，inv（求逆矩阵函数）是内装函数，不能显示。即用户只能使用函数 M 文件，但不能查看函数 M 文件的内容。下面编制一个简单的函数文件供读者参考。

【例 3-2】 编制一个程序，计算三角形的面积 A，已知三角形的 3 条边为 a、b、c，面积计算公式为 $A=\sqrt{s(s-a)(s-b)(s-c)}$，其中 $s=(a+b+c)/2$。

解：

程序如下：

```
function A = triarea(a,b,c)              %程序定义行
if a < 0|b < 0|c < 0                     % 若 a、b、c 为负值，则中断程序，并显示原因
    disp('The a,b,c must be positive number.'),break
end
if a + b < c|a + c < b|b + c < a         % 若三角形中，任意两边之和小于第三边，则中断程序，并显示原因
    disp('This is impossible.'),break
```

```
end
        s = (a + b + c)/2;                                %将三角形周长的1/2赋予s
        A = s* (s - a)* (s - b)* (s - c)^(1/2);           %计算三角形面积
% TRIAREA(a,b,c) is calculate the area of triangle.        %帮助文本及附注
% The a,b,c must a scale and nonnegtive.
% and any two sides of triangle must be great then the third.
% For example triarea(3,4,5)
%          ans =
%              6
%          triarea(3,0,8)
%          ans =
%              0
%          triarea(11,18,6)
%          this is impossible.
%          triarea(-5,6,7)
%          The a,b,c must be positive number.
% See also POLYAREA                                      % 参考函数名，即计算多边形面
                                                           积函数polyarea
```

已知 $a=4$，$b=8$，$c=\sqrt{80}$，求三角形面积。运行程序 triarea

```
>>triarea(4,8,80^(1/2))
ans =
    16
```

已知 $a=4$，$b=8$，$c=14$ 求三角形面积。运行程序 triarea

```
>>triarea(4,8,14)
This is impossible.
```

3.3 内联函数

函数 M 文件是为了计算常用的、需要存储的函数，函数 M 文件的编制要求按 3.2 节 5 条要求进行。但对于一次性使用的函数，可以用内联函数 inline 来实现。内联函数的编制比较宽松，但它不进行函数存储，它的书写格式为

F = inline(expr,'x1','x2',…,'xn')

inline——内联函数；

expr ——以字符串形式的数学表达式；

x1，x2，…，xn——数学表达式 expr 中的变量。

【例 3-3】 用内联函数来表示 $y=\sin x+\sin^2 x$，并求 $x=\pi/4$ 时的函数值。

解：

在命令窗口键入如下程序：

```
>>y = inline('sin(x) + sin(x)^2','x')                   %编制内联函数
y =                                                       %程序响应
```

```
    Inline function:
    y(x) = sin(x) + sin(x)^2
>>y(pi/4)                                     %计算 x = pi/4 时的函数值
ans =
    1.2071
```

【例 3-4】 用内联函数来表示 $y = \sin x_1 \cos x_1 + \sin x_2 \cos x_2 - \sin x_1 \cos x_2$，并求 $x_1 = \pi/8$，$x_2 = 3\pi/8$ 时的函数值。

解：

在命令窗口键入如下程序：

```
>>y = inline('sin(x1)*cos(x1) + sin(x2)*cos(x2) - sin(x1)*cos(x2)','x1','x2')
                                              %输入内联函数
y =
    inline function:                          %程序响应
    y(x1,x2) = sin(x1)*cos(x1) + sin(x2)*cos(x2) - sin(x1)*cos(x2)
>>y(pi/8,3*pi/8)                              %x1、x2 赋值后的函数值
ans =
    0.5607
```

3.4 文本 M 文件

对于处理一个特定任务，一次要执行大量的 MATLAB 命令和语句，且经常重复使用的程序，则可将这些语句的集合，存放在扩展名为 M 的文件中。M 文件的编辑，与函数 M 文件相类似，在 M 文件编辑器中进行，它执行计算所用的数据来自键盘输入和工作空间中已经存在的变量。它的输出数据亦保存在工作空间中，便于下次计算时的调用。为了与函数文件的区别，故称为文本文件。函数 M 文件与文本 M 文件的区别有以下几点：

(1) 函数 M 文件在函数名中接受输入数据，而文本 M 文件只向工作空间或键盘接受数据。

(2) 函数 M 文件的运算操作在内部专用存储器中进行，与工作空间无关，而文本 M 文件的数据交换则在工作空间进行。

(3) 文本 M 文件能调用函数 M 文件，而函数 M 文件不能调用文本 M 文件，否则将显示出错。

下面显示一个 M 文件用来绘制 4 幅不同的花瓣图，通过 Enter 键来切换。

【例 3-5】 在单位圆内，分别绘制 10 花瓣、20 花瓣、2 花瓣和 2-3-2 不对称花瓣图。通过 Enter 键来切换图形。

解：

在 M 文件编辑器输入如下程序：

```
%flower petal plots                           %程序标题，花瓣图
theta = -pi:pi/300:pi;                        %设置角度向量，分度为pi/300
rho(1,:) = 2*sin(5*theta).^2;                 %计算频率为 5 的正弦函数平方
rho(2,:) = cos(10*theta).^3;                  %计算频率为 10 的余弦函数立方
```

```
rho(3,:) = sin(theta).^2;
rho(4,:) = 5* cos(3.5* theta).^3;             %计算频率为 3.5 的余弦函数立方
for i = 1:4                                   %设置循环次数
    polar (theta, rho (i,:))                  %绘制上述函数的极坐标图
    switch i                                  %根据不同的 i 的值，写入不同的标题 Swith
                                                为开关语句，详细说明见下几节
    case 1
        title 10-petal                        %标题为 10 花瓣
    case 2
        title 20-petal                        %标题为 20 花瓣
    case 3
        title two-petal                       %标题为 2 花瓣
    case 4
        title 2-3-2-unsymmetrical-petal       %标题为 2-3-2 不对称花瓣
    end
    pause                                     %程序暂停，当按 Enter 键，则继续执行下一
                                                个循环
end                                           %程序结束
```

在 MATLAB 命令窗口输入

```
≫flower_plots
```

则程序开始运行，并得出如图 3-2 ~ 图 3-5 共 4 幅花瓣图。该程序调用的函数有 sin、cos、polar 以及流程控制命令 for/end、switch/end、pause/enter 等。

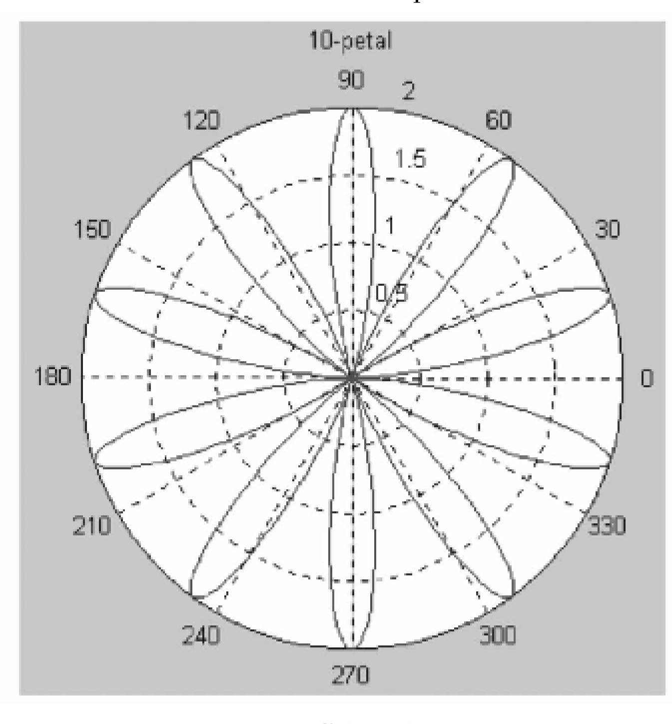


图 3-2　花瓣图之一

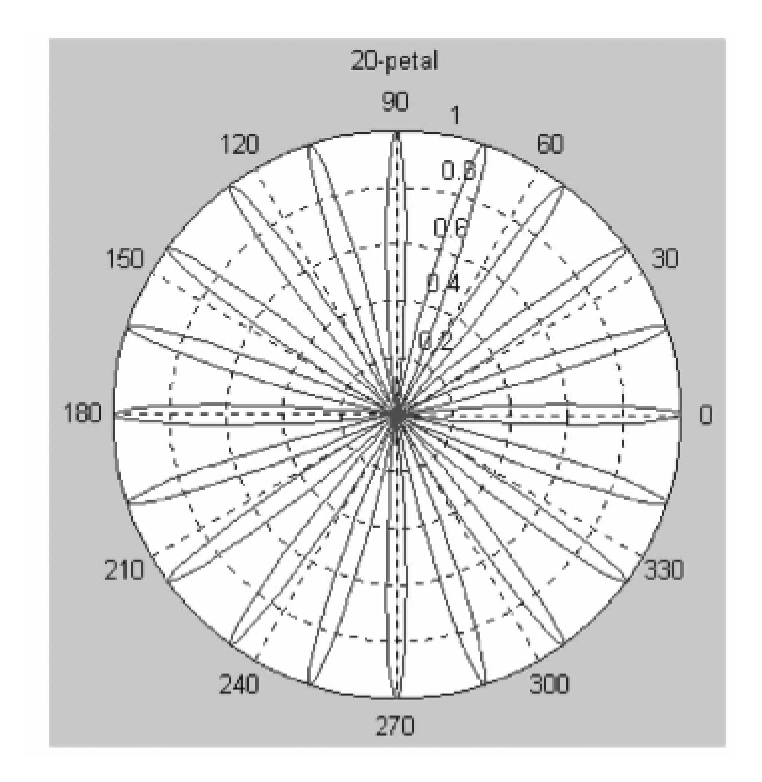


图 3-3 花瓣图之二

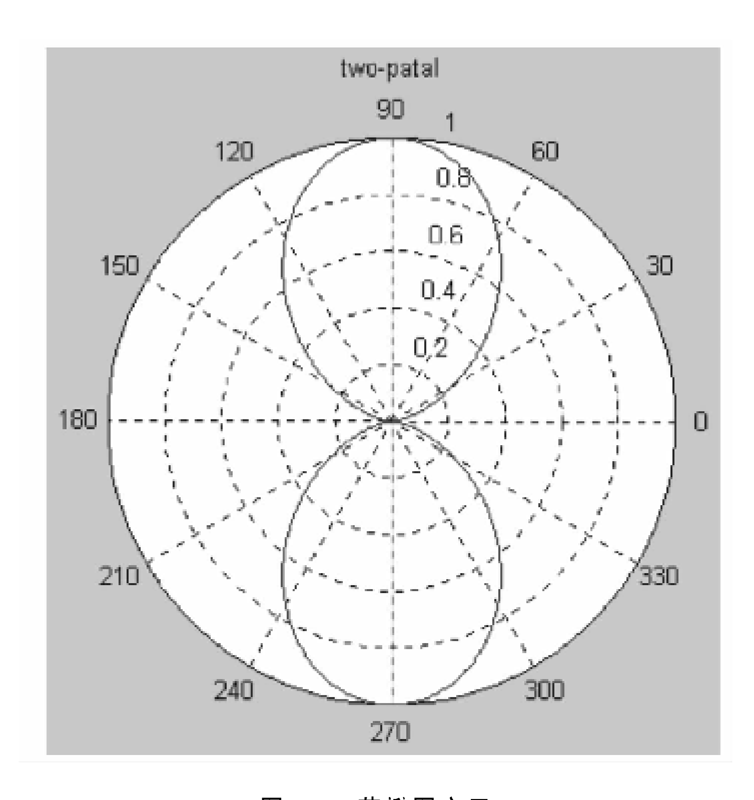


图 3-4 花瓣图之三

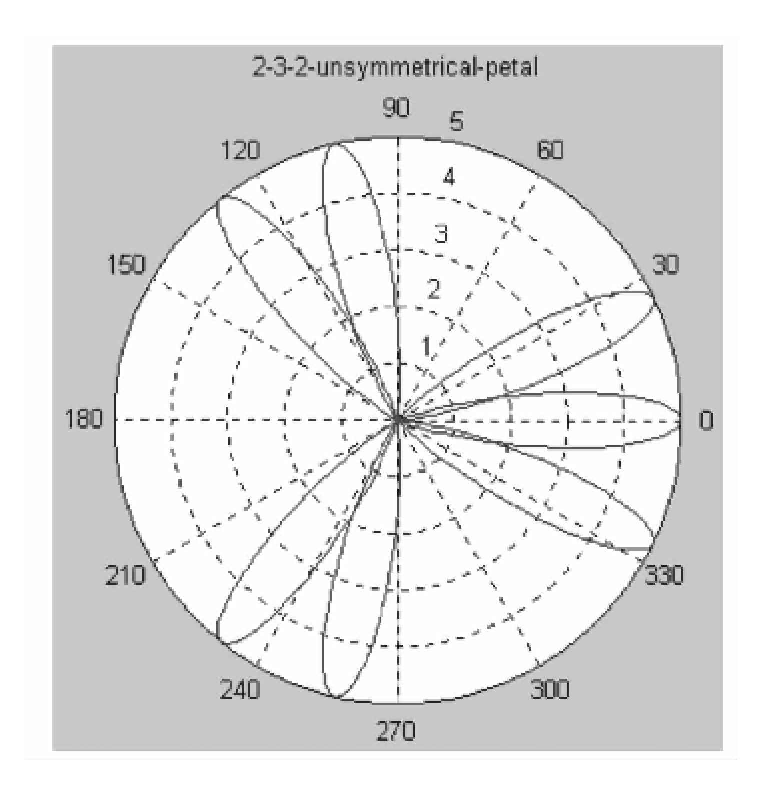


图 3-5　花瓣图之四

3.5　M 文件的编辑和存储

M 文件的编辑，点击菜单栏中 file/new/M-file 后即出现 M 文件编辑器。函数 M 文件即可在 M 文件编辑器中，按照 3.2 节 5 个部分要求进行编辑。编辑过程中需要进行调试和排除错误。当确认无误时，再进行存盘，存盘时点击工具栏中存盘命令，文件名是自动生成的，即原先设置的函数名。

对于文本 M 文件的文件名由程序编写者自行命名，程序的组成也比较自由。对程序的调试和排除错误，使程序简洁和易读，缩短程序执行的时间，则是对编程的共同要求。

为了对 M 文件的编写，必须熟悉 MATLAB 的流程控制。MATLAB 流程控制，有循环控制语句 for/end、while/end；条件控制语句 if/else/and；分支条件控制语句 if/elseif/ elseif/…else/end；开关控制语句 switch/case 1，cause 2… otherwise/end；试验和捕捉语句 try/catch/end 和返回语句，return 中断语句；break 等。以下分别予以介绍，并给予举例说明。

3.6　循环控制语句之一：for/end

for/end 是用于要求重复多次执行 for 与 end 之间的程序语句，它允许嵌套使用。其书写格式如下：

```
for 变量 = 起点:增量:终点          %程序的重复次数由变量的起点、终点和增量来决定
    程序语句
end                                %与 for 循环相对应的语句
```

【例 3-6】　计算 8 阶 Pascal 矩阵的所有元素的总和。

解：

```
>>A = pascal(8);                                  %设矩阵 A 为 Pascal 8 阶矩阵
>>s = 0;                                          %设数值变量初值 s 为零
>>for i = 1:8                                     %设置 i 为 8 次循环
      s = s + sum(A(i,:));                        % 计算 A 矩阵的第 i 列元素的和，
                                                    sum 为对矩阵 A(i,:)的 i 行元素
                                                    求和的命令
  end, s                                          %结束循环后，显示总和 s
s =
      12869
```

本例只说明 for/ end 的应用。其实有更简单的计算方法如下：

```
>>sum(sum(pascal(8)))                             %计算 Pascal 矩阵的各列向量之和，
                                                   再求总和，其结果与上面编程计
                                                   算相一致
ans =
      12869
```

【例 3-7】 用 Gauss 消去法使已知 5 阶魔方矩阵变换为上三角矩阵。

解：

在 MATLAB 命令窗口输入如下程序：

```
>>A = magic(5)                                    %设矩阵 A 为 5 阶魔方矩阵
A =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9

>>for j = 1:4                                     %设置列向操作次数，1~4 列
     for i = j + 1:5                              %设置行向操作次数，2~5 行
         A(i,:) = A(i,:) - A(i,j)/A(j,j) * A(j,:); %对矩阵 A 下三角进行消元操作
     end                                          %结束 i 循环
  end, A                                          %结束 j 循环，并显示消元后的结果
A =                                               %经消元后的上三角矩阵

   17.0000   24.0000    1.0000    8.0000   15.0000
         0  -27.4706    5.6471    3.1765   -4.2941
         0         0   12.8373   18.1585   18.4154
         0         0         0   -9.3786  -31.2802
         0         0         0         0   90.1734
```

3.7 循环控制语句之二：while/end

while/end 与 for/end 不同，它不规定循环次数，而由 while 后面的条件表达式来决定，若条件成立，则重复执行 while 与 end 之间的程序语句，否则结束循环。其循环格式如下：

```
while 条件表达式
    程序语句
end
```

【例 3-8】 用迭代法求解 $x^3-3x^2-5=0$ 的一个实根。

解：

把方程式改写成迭代形式，即 $x=3+5/x^2$，并设初始值 $x_0=3$，解题程序如下，程序名为 itersolve1：

```
x0 = 3; r = 1; n = 0;           %初值设置，r 为计算误差初值，n 为迭代次数
while r > 1e-5                  %当迭代误差小于设定误差 1e-5 时，结束循环
    x1 = x0;                    %上次计算值代入下一次
    x0 = 3 + 5/x1.^2;           %迭代运算
    r = abs(x0 - x1);           %误差计算
    n = n + 1;                  %迭代次数加 1
end, x0, n                      %循环，循环结束后显示 x0 和 n
>>itersolve1                    %运行程序
x0 =
    3.4260
n =
    9
```

如果直接使用多项式求根命令，则得

```
>>roots([1 -3 0 -5])            %roots 为多项式求根命令，方括号内为求解方程式的系
                                 数向量
ans =                           %求根结果
    3.4260                      %实根，与上述计算相同
   -0.2130 + 1.1891i            %共轭复根
   -0.2130 - 1.1891i
```

【例 3-9】 用迭代法求解下列线性方程组：

$$\left.\begin{aligned}7x_1-3x_2+2x_3&=17\\4x_1+9x_2-x_3&=29\\6x_1+3x_2+11x_3&=35\end{aligned}\right\}\tag{3-1}$$

解：

将上述方程组改成迭代形式

$$\left.\begin{aligned}x_1&=3x_2/7-2x_3/7+17/7\\x_2&=-4x_1/9+x_3/9+29/9\\x_3&=-6x_1/11-3x_2/11+35/11\end{aligned}\right\}\tag{3-2}$$

由此写成矩阵迭代形式为

$$X_1 = A_1 X_0 + B_1$$

为了执行迭代运算在命令窗口输入如下程序：

```
>>A1 = [0 3/7 -2/7; -4/9 0 1/9; -6/11 -3/11 0]   %迭代形式的系数矩阵
A1 =

         0    0.4286   -0.2857
   -0.4444         0    0.1111
   -0.5455   -0.2727         0

>>B1 = [17/7;29/9;35/11]                          %迭代形式的常数项列向量
B1 =

    2.4286
    3.2222
    3.1818

>>X0 = [0 0 0]'; X1 = [1 1 1]'; n = 0;            %设置迭代向量的初值 X0，X1，n 为循环次数
>>while norm(X1 - X0) > 1e-5                      %当差向量的范数小于 1e-5 时，终止循环，否则执行循环体
        X1 = X0;                                  %上次计算值代入下一次
        X0 = A1*X1 + B1;                          %迭代运算
        n = n + 1;                                %循环次数加 1
  end, X0, n                                      %循环结束，显示迭代结果和迭代次数
X0 =                                              %迭代结果
   3.0000
   2.0000
   1.0000
n =                                               %迭代次数
   17
>>A = [7 -3 2;4 9 -1;6 3 11]                      %式（3-1），系数矩阵
A =

     7    -3     2
     4     9    -1
     6     3    11

>>B = [17;29;35];                                 %式（3-1），常数项向量
>>X = inv(A)*B;                                   %用逆矩阵求解式（3-1）的结果与迭代解法的结果完全一致
>>X'
ans =
```

3　　　2　　　1

3.8 分支条件选择语句 if/end

if 语句是判定一个逻辑表达式，由逻辑表达式的值来决定是否执行它下面的一组程序语句。在最简单的情况下，它的语法结构如下：

if 逻辑表达式

　程序语句

end

假如逻辑表达式是真（逻辑值为 1），则 MATLAB 执行 if 和 end 之间的程序语句，并继续执行 end 以后的程序。若逻辑表达式是假（逻辑值为 0），则跳过 if 与 end 之间的程序语句，继续执行 end 以后的语句。

下面用简单例子来说明：

【例 3-10】 输入一整数，若能被 2 除尽，则显示它为偶数并显示除 2 后的商，否则显示此数为奇数。

解：

在 M 文件编辑器中输入如下程序（程序名为 examp3 _ 8）：

```
    %本程序判断输入整数是偶数还是奇数
x = input('input a integer \n')      %输入整数
if rem(x,2) == 0                     %假如 x 能被 2 整除，则余数等于零，rem(a,b)为求 a/b
                                       所得余数的函数
    disp('x is even')                %显示 a 是偶数，disp 为显示字符串语句
    b = x/2                          %计算除 2 后的商
    fprintf('%d \n',b)               %输出b,fprintf 为写格式数据语句
else disp('x is odd')                %否则，显示 x 是奇数
end
```

在命令窗口运行上述程序如下：

```
>>example3 _ 8
input a integer
235
x =
    235
x is odd
>> examp3 _ 8
input a integer
256
x =
    256
x is even
b =
```

128

if 语句在程序中可以嵌套任意次。对于 if 语句后的逻辑表达式，如果判断一个非标量，那么它的所有元素必须为非零，才认为是真。例如，若 X 是矩阵，程序为

```
if   X
    程序语句
end
```

那么它相当于

```
if   all(X(:))
    程序语句
end
```

3.9 多分支条件选择语句 if/elseif/…/else/end

elseif/else 是进一步的 if 条件语句，其结构形式如下：

```
if 逻辑表达式 1
  程序语句 1
elseif 逻辑表达式 2
      程序语句 2
elseif 逻辑表达式 3
      程序语句 3
      ……
else
    程序语句 n
end
```

在这种格式中，if 与 elseif 是互相并列（即互相独立且互相排斥的）条件选择语句，若某个逻辑表达式是真，则程序执行相应的程序语句，执行后，则继续 end 以后的程序语句，否则，则跳过相应的程序语句，继续下一个 elseif 的逻辑判断。唯有 else 是无逻辑表达式，它执行的条件是与先前所有的逻辑判断条件相否。现举例如下：

【例 3-11】 用二分法求多项式的根。已知多项式为

$$f(x)=x^3-3x^2+5x-7=0 \tag{3-3}$$

它在区间[0,4]中有一实根，且 $f(0)<0, f(4)>0$ 求此实根，允许误差为 1E－5。

解：编制用二分法解代数方程式的根的程序文件 examp3 _ 9 如下：

```
%用二分法求多项式的实根
while abs(b-a) > tol              %设置初值[a,b]及容许误差 tol
    x = (a+b)/2;                  %用二分法作试算值
    fx = fun _ 101(x);            %调用代数方程式的函数文件，fun _ 101 为函数名
    if fx < 0                     %若 fx 小于零，则缩小负值搜索区
        a = x;
        else b = x;               %否则，缩小正值搜索区
    end
```

```
end, x                                        %结束 while 循环，并显示方程式根 x
```

编制已知代数方程式（3-3）的函数文件为 fun _ 101 程序如下：

```
function y = fun_101(x)
y = x.^3 - 3*x.^2 + 5*x - 7;
```

在命令窗口输入初值a，b 和允差 tol，并运行用 examp3 _ 9 结果如下：

```
>>a = 0;  b = 4;  tol = 1e-5;
>>examp3_9
x =
    2.1795
```

如果用多项式求根的命令求解，则首先确定多项式的系数向量 V，随后用roots(V)求解如下：

```
    >>V = [1, -3, 5, -7];                    %由式（3-3）得到的系数向量
    >>roots(V)
    ans =
      2.1795
      0.4102 + 1.7445i
      0.4102 - 1.7445i
```

这个计算结果与二分法求实根的结果相一致。但使用 roots 求根，它把所有的根都求出来了。

3.10 开关语句 switch/end

开关语句 switch 是一种多分支选择语句，它执行指定程序语句，取决于变量或表达式的值。其格式如下：

```
switch 表达式
   case 值 1
       程序语句                              %假若表达式的值是值 1，则执行
   case 值 2
       程序语句                              %假若表达式的值是值 2，则执行
   case 值 3
       程序语句                              %假若表达式的值是值 3，则执行
   ……
   otherwise
       程序语句                              %假若表达式的值是其他值，则执行
   end                                       %结束开关语句
```

在有些情况下 switch 语句也可以用条件分支语句来代替。

【例 3-12】 用二次方程式的判别式来判别根的情况。

解：

程序如下（程序名为 examp3 _ 10）：

```
%用判别式 d = b^2 - 4*a*c，分析二次方程式的根的性质
```

```
delta = input('input coefficient of square equation as [a b c],\n')   %输入系数向量 delta
switch sign(delta(2)^2 - 4* delta(1)* delta(3))                       %判别 b^2 - 4* a* c 的符号
case 1                                                                %判别式大于零
    disp('have two unequal real roots')                               %有两个不等实根
case 0                                                                %判别式等于零
    disp('have two equal real roots')                                 %有两个相等实根
case-1                                                                %判别式小于零
    disp('have two complex roots of conjugate')                       %有一对共轭复根
otherwise                                                             %否则
    disp('may be error')                                              %可能出错
end                                                                   %结束开关语句
```

在命令窗口运行该程序得结果如下：

```
>>examp3_10                                                           %运行上述程序
input coefficient of square equation as [a b c],                      %提示输入数据
[1 2 5]                                                               %输入二项式系数
delta =                                                               %程序响应
     1     2     5
have a pair of complex roots                                          %有一对复根
```

3.11 出错处理语句 try/catch/end

在编写程序中，如果估计到可能出现的错误，一旦发生错误，使程序转向另一程序段，使程序能继续进行。在这种情况下，程序中可插入 try/catch/end（试验/捕捉）语句，其格式如下：

```
try
程序语句 1
catch
程序语句 2
end
```

在正常情况下，程序执行 try 与 catch 之间程序语句 1 以及 end 以后的程序语句。当程序语句 1 出现错误时，本来会终止程序的执行，但当插入 try/catch 语句以后，程序进入 catch/end 之间的程序语句 2 以及 end 以后的程序语句。现举例如下：

【例 3-13】 已知矩阵 $\boldsymbol{A}$ 和列向量 $\boldsymbol{b}$，求线性方程组 $\boldsymbol{Ax}=\boldsymbol{b}$ 的解。当方程式数 m 小于变量数 n，则显示有无穷多个解。当 $m=n$，则用程序x = inv(A)* b 求解。当 $m>n$ 时则用程序x = pinv(A)* b 求解。

解：

共程序如下：

```
% This program for solution lineal equations
clear                                                                 %清除变量
A = input('输入系数矩阵\n')              %输入系数矩阵 A
```

```
b = input('输入等式右侧列向量\n')      %输入常数项列向量 b
[m,n] = size(A);                       %测试行数、列数
m1 = size(b);
if m ~ = m1                            %假若 m、m1 不等，则中断程序
      disp('A,b 的行数应相等')
      break
end
if n > m                               %假若 n > m，则显示有无穷个解，并且中断程序
      disp('有无穷多个解')
      break
end
r = rank(A)                            %测试矩阵 A 的秩
try                                    %插入试算
   x = inv(A) * b                      % 以矩阵 A 求逆运算，若出现错误，则转向 catch 与
                                         end 之间的程序语句

catch
      x = pinv(A) * b                  %用广义逆矩阵求解
      lasterr                          %lasterr 为显示出错信息
end                                    %结束试验/捕捉语句
```

程序运行如下：

```
≫examp3 _ 11                           %运行上述程序

输入系数矩阵
magic(3)
A =
      8      1      6
      3      5      7
      4      9      2
输入等式右侧列向量
[1;2;3]
b =
      1
      2
      3
r =                                    %矩阵 A 的秩
      3
x =                                    %程序中未出现错误时，方程式的解
      0.0500
      0.3000
```

```
    0.0500
>>example3_11                          %再次运行上述程序，但方程式数大于变量数的情况
输入系数矩阵
[5 1 6;7 2 8;9 6 4;9 1 1]
A =
       5       1       6
       7       2       8
       9       6       4
       9       1       1
    输入等式右侧列向量
[4.9;6.9;7.1;3.6]
b =
    4.9000
    6.9000
    7.1000
    3.6000
r =                                    %矩阵 A 的秩
    3
x =                                    %秩数小于行数，用逆矩阵求解时出现错误，所以程序改用广义逆矩阵求解，这时的解，并非精确解，而是满足最小二乘解
    0.3000
    0.4000
    0.5000
ans =                                  %显示错误原因，用逆矩阵求解时，矩阵应为方阵
Error using ==> inv
Matrix must be square.
```

3.12 continue、break 和 return 语句

continue 语句经常与循环语句 for/end、while/end 结合使用。当循环语句中出现 continue 时，则不再继续执行当前循环体现内程序语句，即提前结束当前循环，并继续执行下一个循环。现举例如下：

【例 3-14】 在整数 2～100 中，找出不能被 2、3、5、7、11 整除的数。

解：

编制 M 文件如下（程序名为 examp3_12）：

```
%本程序实现在 2～100 整数中寻找不能被 2、3、5、7、11 整除的数
i = 1;                                 %设初值
for n = 2:100                          %设 for 循环
```

```
if mod(n,2) ==0|mod(n,3) ==0|mod(n,5) ==0|mod(n,7) ==0|mod(n,11) ==0
                                    %若 n 能被 2、3、5、7、11 除尽则滑过，其中 mod 为求
                                      余数函数
    continue
else                                %否则将 n 赋值给向量 X
    X(i) = n;i = i + 1;
    end
end
X                                   %显示结果
```

在命令窗口运行该程序如下：

```
≫examp3 _ 12
X =
    Columns 1 through 15
      13    17    19    23    29    31    37    41    43    47    53    59    61
      67    71
    Columns 16 through 20
      73    79    83    89    97
```

break 语句与 continue 语句相似，但它不是继续执行下一次循环，而是退出循环，并继续执行循环以外的程序。举例如下：

【例 3-15】 用迭代法解下列多项式的实根。

$$x^3 - 5x + 5 = 0 \tag{3-4}$$

解：

编制程序如下（程序名为 examp3 _ 13）：

```
%本程序用迭代法解多项式 x^3 -5x +3 =0 的根
x0 = input('输入实根的初始值\n')
r = input('输入允许误差\n')
for n  =1:50                        %设置 for 循环
     x1  = (x0.^3 + 3)/5;           %将多项式改写为迭代形式
       if abs (x1 - x0) < r         %若误差小于设定值，则中断程序
           break
       else x0 = x1;                %否则进行迭代运算
       end
end
fprintf('方程式的根为:\n')
x1
fprintf('迭代次数\n')
n
```

在命令窗口运行上述程序

```
≫examp3 _ 13
```

输入实根的初始值
1.5
x0 =
    1.5
输入允许误差
1e－5
r =
    1.0000e－005
方程式的根为：
x1 =
    0.6566
迭代次数
n =
    12

图 3-6 显示用迭代法解方程式（3-4）的过程。x0 的初始值为 1.5，经过 12 次迭代，得到误差小于万分之一的解为 0.6566。

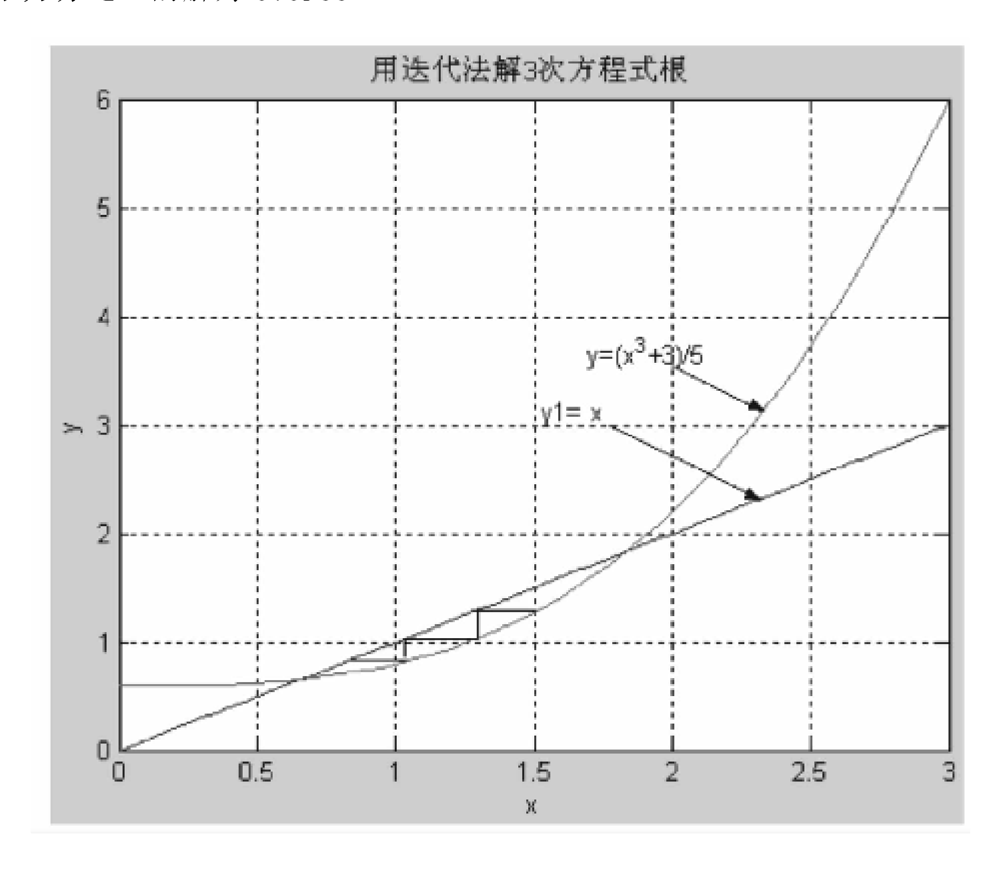


图 3-6　用迭代法解 3 次方程式的实根

return 语句用于对调用它的函数的正常返回。在正常情况下，当被调用函数到达程序末尾时，它是自动返回的。但若在程序中加入 return 语句时，可以强制它提前返回。return 语句也用于终止键盘工作方式。

【例 3-16】　在命令窗口输入矩阵 A，计算它的行列式，若 A 为空阵，则定义它为 1，

并返回调用函数。

解：

求解行列式的函数名为 det，其程序如下：

```
function d = det(A)                    %函数名
    if isempty(A)                      %若为空阵，则赋值为 1，并提前返回
        d = 1;
        return
    else                               %否则进入计算行列式的程序体
        [m,n] = size(A);
        …
    end
```

在命令窗口输入空阵得行列式为 1

```
>>A = [];
>>det(A)
ans =
    1
```

在命令窗口输入 3 阶魔方矩阵得行列式为 -360

```
>>det(magic(3))
ans =
   -360
```

3.13 奇数阶魔方矩阵的编程

通过对 MATLAB 编程命令的介绍，我们来计算奇数阶魔方的程序编写。奇数阶魔方的设计方法是多种多样的。MATLAB 中的魔方矩阵的程序是隐藏的。若用 type 命令显示 magic.m 时，它会显示这是内装程序，无法展现。魔方矩阵的设计思路之一，是构造一个扩展的零矩阵 A（参见图 3-7，3 阶魔方示意图），随后用顺序数矩阵 v 填入 A 的相应对角线，再用行、列叠加使矩阵 A 的中心 n 阶矩阵零元素填满。最后取出中心 n 阶方阵，即得魔方矩阵。

图 3-7 三阶魔方创建示意图

下面是奇数阶魔方矩阵的程序，程序名为 odd_magic。

```
% This is program for create a matrix of odd order magic
clear                                  %清内存
n = input('please input a odd number \n')   %输入一个奇数
if mod(n,2) == 0                       %若为偶数则中断程序
    'input number must be odd.',break
end
n1 = 2* n - 1; m = (n - 1)/2;          %计算扩展零矩阵 A 的行、列数 n1 和上、下
                                         对角线数 m(不包含主对角线)
A = zeros(n1);
for i = 1:n                            %设置顺序数矩阵 v
```

```
    for j = 1:n
        v(i,j) = (i-1)*n+j;
    end
end
for k = 1:m+1                              %在 A 矩阵内，填写 m+1 条上对角线
    B = zeros(1,n-2+2*k);
    B(k:(k+n-1)) = v(k,:);
    A = A + diag(B,n+1-2*k);
end
for k = m+2:n                              %在 A 矩阵内，填写 m 条下对角线
    B = zeros(1,3*n-2*k);
    B(n-k+1:2*n-k) = v(k,:);
    A = A + diag(B,n+1-2*k);
end
if m == 1                                  %3 阶魔方时，行、列叠加填数
    A(4,:) = A(4,:) + A(1,:);A(2,:) = A(2,:) + A(5,:);
    A(:,4) = A(:,4) + A(:,1);A(:,2) = A(:,2) + A(:,5);
else                                       %3 阶以上奇数阶次时，行、列叠加填数
    Bu = A(1:m,:);
    Bd = A((n1-m+1):n1,:);
    Bl = A(:,1:m);Br = A(:,(n1-m+1):n1);
    A(m+1:2*m,:) = A(m+1:2*m,:) + Bd;
    A((n1-2*m+1):n1-m,:) = A((n1-2*m+1):n1-m,:) + Bu;
    A(:,m+1:2*m) = A(:,m+1:2*m) + Br;
    A(:,(n1-2*m+1):n1-m) = A(:,(n1-2*m+1):n1-m) + Bl;
end
M = A(m+1:(n1-m),m+1:(n1-m))               %取中央 n 阶方阵，即为魔方矩阵
```

【例 3-17】 用上述程序，计算 3 阶魔方矩阵。

解：

在命令窗口输入如下程序：

```
>>odd_magic                                %调用奇数阶魔方程序
please input a odd number                  %输入阶次
3
n =
    3
M =                                        %3 阶魔方矩阵
    4    9    2
    3    5    7
```

```
      8         1         6
```

【例 3-18】 用上述程序，计算 5 阶魔方矩阵。

解：

在命令窗口输入

```
   ≫odd_magic                              %调用奇数阶魔方程序
please input a odd number                  %输入阶次
5
n =
     5
M =                                        %5 阶魔方矩阵
    11    24     7    20     3
     4    12    25     8    16
    17     5    13    21     9
    10    18     1    14    22
    23     6    19     2    15
```

为了了解魔方矩阵的建立过程，查看下列各矩阵

```
≫v                                         %顺序数矩阵
v =
     1     2     3     4     5
     6     7     8     9    10
    11    12    13    14    15
    16    17    18    19    20
    21    22    23    24    25
```

查看填写顺序数方阵后，观察行、列叠加后的 A 矩阵

```
≫A                                         %扩展矩阵
A =
    0     0     0     0     1     0     0     0      0
    0     0     0     6     0     2     0     0      0
    0     0    11    24     7    20     3     0      0
    0    16     4    12    25     8    16     4      0
   21     0    17     5    13    21     9     0      5
    0    22    10    18     1    14    22    10      0
    0     0    23     6    19     2    15     0      0
    0     0     0    24     0    20     0     0      0
    0     0     0     0    25     0     0     0      0
```

取矩阵 A 中心的 5 阶方阵即为所求 5 阶魔方矩阵。

调用 MATLAB 的 5 阶魔方矩阵如下：

```
≫magic(5)
ans =
```

```
    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22
    10    12    19    21     3
    11    18    25     2     9
```

它与 M 矩阵的排列不同，但行、列元素之和与对角线之和是相同的，因为

```
≫sum(M)                                    %列元素之和的向量
ans =
    65    65    65    65    65
≫sum(M′)                                   %行元素之和的向量
ans =
    65    65    65    65    65
≫trace(M)                                  %主对角线元素之和
ans =
    65
≫trace (M′)                                %反向主对角线元素之和
ans =
    65
```

3.14 数据类型概述

MATLAB 建立了多种数据类型来满足不同用户的需要。MATLAB 有 14 种基本数据类型（或称等级）。如字符型、整数 8 位、整数 16 位、整数 32 位、数值 8 位、数值 16 位、数值 32 位、数值单精度、数值双精度、稀疏型、单元型、结构型、java 型和句柄函数。每一种数据类型都是以数组方式出现。这些数组最小为零维数组，而且能扩展成 n 维数组。矩阵则被称作为二维数组。

所有这些数据类型是显示在图 3-8 数据分类的树干图上。作为补充，用户定义的数据类型作为结构数组的子集。

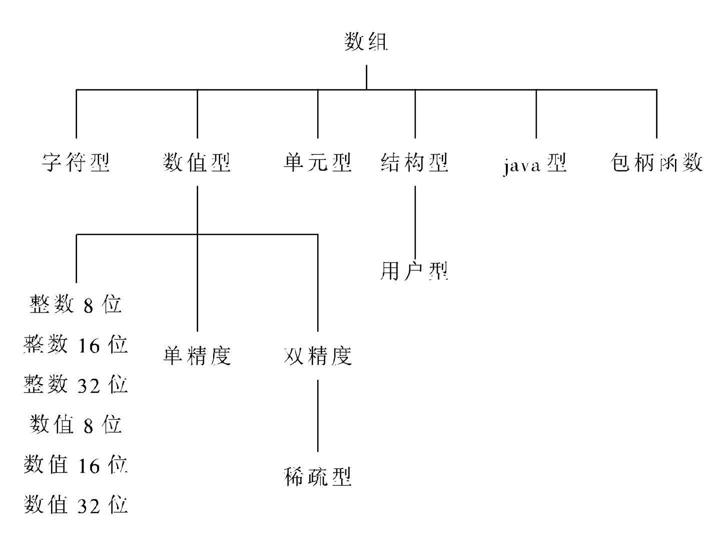


图 3-8 数据分类的树干图

字符型数据是由单引号括起来的字符串或与数字组合的字符串组成。它的存储方式是以ASCII码存储。一个字符串在MATLAB中是作为一维数组，它的长度恰好等于字符的个数。字符型数组通常用于程序的操作提示、文字段的搜索或程序输出结果的说明等。字符型数组可以进行串联、并联连接，字符比较，字符搜索和替换等。

数值型数组包括有符号与无符号的整数，单精度和双精度的浮点数以及双精度的稀疏矩阵。下面几点在MATLAB中，对数值型数据是有效的：

(1) 所有MATLAB计算是以双精度进行计算的。

(2) 对整数及单精度数组比双精度数值提供更多的存储器，更有效地进行存储。

(3) 所有的数据类型支持基本数据操作。如数组的下标引用和数组的重写。

(4) 为了执行整数和单精度数组的数学运算，必须用函数double将它们转换成双精度。

单元数组提供存储机构，可以存储不同种类型或不同大小的数组在单元数组的单元里，例如可以存储1行50字符的数组，7行13列的双精度数组和1个32位二进制数到单元数组的单元里。当需要访问单元数组时，可以用访问矩阵（或数组）同样的方法来操作。

在MATLAB中，结构数组与单元数组相似，它也能存储不同类型的数据，但是在这种情况下，结构数据的数据场的名称胜过单元数组。因为结构数组中的数据，贴有结构数组的场名。当访问结构数据时，用同样的数据场名。

MATLAB数据是分级的。用户可以建立自己的用户定义级（即用户型），用户定义级是被安置在MATLAB的结构数组下面，它是结构数组的子集。如前面提到的数组分类的树干图所描述。

3.15 字符型数组

用单引号括起来的字符和数字的组合称为字符串或称为字符数组。MATLAB中对字符串的书写格式为

S = 'Any Charactiors'　　(3-5)

S = char (x)　　(3-6)

X = double (S)　　(3-7)

式(3-5)用来创建任意字符的字符串。

式(3-6)是用ASCII代码的向量x，来创建字符串。

式(3-7)则是将字符串转换成MATLAB的双精度数。应该指出，字符串与字符数组是相同的概念。

字符数组与矩阵一样可以实行水平连接（strcat）和垂直连接（strvcat），可以进行字符数组的比较（strcmp），寻找（findstr），字符串替换（strrep）等运算。下面举例予以说明。

【例3-19】 建立字符串数组A = 'Today is Saturday.'及B = 'I want go home.'检查它们的长度，将它们进行水平连接和垂直连接。

解：

```
>>A = 'Today is Saturday.'                %输入字符数组 A
A =
Today is Saturday.
```

```
≫B = 'I want go home.'                          %输入字符数组 B
B =
I want go home.
≫na = size(A)                                   %检查 A 的长度
na =
      1     19
≫nb = size(B)                                   %检查 B 的长度
nb =
      1     15
≫AB = strcat(A,B)                               %字符数组 A、B 进行水平连接
AB =
Today is Saturday. I want go home.
≫ab = strvcat(A,B)                              %字符数组 A、B 进行垂直连接
ab =
Today is Saturday.
I want go home.
≫size(ab)                                       %垂直连接后的大小
ans =
      2     19
≫double(ab)                                     %转换成 ASCII 码，其中代码 32 为
                                                  空格
ans =
  Columns 1 through 17
      84    111    100     97    121     32    105    115     32    115     97    116
117    114    100     97    121
      73     32    119     97    110    116     32    103    111     32    104    111
109    101     46     32     32
  Columns 18 through 19
      46     32
      32     32
```

在 MATLAB 中寻找已知字符串 str1 中是否含有字符串 str2，的函数为

$$k = \text{findstr}('str1','str2') \tag{3-8}$$

式中，k 为返回向量，k 指出 str2 的起始位置。如果字符串 str1 中第 n 个字符开始有字符串 str2，则 k 为 n。

若 str1 与 str2 位置互换则对 k 无影响。

【例 3-20】 已知字符串 s = 'Find the starting indices of the shorter string.'；寻找单词“the”的起始位置。

解：

```
≫s = 'Find the starting indices of the shorter string.';     %输入字符串 s
```

```
≫k = findstr(s,'the')                         %寻找字符串 str2，在 s 中的起始位置
k =
     6     30
≫k1 = findstr(s,'student')                    %s 中没有 "student" 这一单词，则
                                                返回空阵
k1 =
     []
≫k2 = findstr('the',s)                        %输入字符串互换，对结果无影响
k2 =
     6     30
```

字符串替换函数为

$$S = strrep(S1, S2, S3) \tag{3-9}$$

式中，S2 为 S1 中含有的字符串，S3 为替换 S2 的字符串，S 为替换后的字符串。

【例 3-21】 已知字符串 S1 = 'The area is at least 400 square meters'，将其中 400 改为 1000。

解：

```
≫S1 = 'The area is at least 400 square meters';    %输入字符串 s
≫newarea = strrep(S1,'400','1000')                 %字符串替换，400，改为 1000
newarea =                                           %替换后的字符串
The area is at least 1000 square meters
```

【例 3-22】 将 ASCII 字符以 3 行 32 列显示。

解：

```
ASCII = char(reshape(32:127,32,3)')
≫asii = char (reshape(32:127,32,3)')     % 将 32 到 127 的数，所组成的数组，
                                           改写成 3 行 32 列，用 ASCII 码显示
assai =                                   %显示 ASCII 码的字符
!"#$%&'()*+,-./0123456789:;<=>?
@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_
`abcdefghijklmnopqrstuvwxyz{|}~
```

3.16 单元数组

单元数组是一种 MATLAB 数组，它的每一元素是单元，在单元里可以保存各种类型的 MATLAB 数组。例如单元数组、实数矩阵、文本字符串数组、结构数组以及其他复数向量等。表 3-1 是 2 行 3 列单元数组的例子。

3.16.1 单元数组的创建

可以通过对单元数组的赋值来创建单元数组，也可以对单元数组的预分配来创建。对于单元数组预分配的书写格式为

$$C = cell(n) \tag{3-10}$$

表 3-1　2 行 3 列单元数据例

Cell 1.1	Cell 1.2	Cell 1.3
8　1　6 3　5　7 4　9　2	'John Smith' '3/8/2005' 'Class II'	2+i　1-2*i 1+2*i　2-i
Cell 2.1	**Cell 2.2**	**Cell 2.3**
[1.4 2.8 3.3 5.6]	2　3　4 3　6　10 4　10　20	'text'　4　2 　　1　5 [4 2 7]　0.5+6*i

$$C = \text{cell}(m, n) \tag{3-11}$$

$$C = \text{cell}(\text{size}(A)) \tag{3-12}$$

式（3-10）为建立 n 阶单元空方阵。

式（3-11）为建立 m 行 n 列的单元空矩阵。

式（3-12）为建立与矩形 A 同维的空单元矩阵。在单元空矩阵确定后，就可以对单元元素的赋值进行操作。

单元数组的赋值可以通过两种方式来进行。一种为

（1）下标索引。与数组表示方法相类似，用圆括号内的下标数来显示单元号。在等式右端用花括号把单元的内容括起来。例如一个 2 行 2 列的单元数组，赋值如下：

```
>>A(1,1) = {magic(3)};                        %在（1，1）单元里为3阶魔方矩阵
>>A(1,2) = {'10/1/2004'};                     %在（1，2）单元里为日期
>>A(2,1) = {1:5};                             %在（2，1）单元里，是1:5的行向量
>>A(2,2) = {[1+2*i 2-3*i;2+3*i 1-2*i]};       %在（2，2）单元里，是复数矩阵
>>A                                           %显示单元数组的结构
A =
      [3x3 double]        '10/1/2004'
      [1x5 double]        [2x2 double]
>>celldisp（A）                               %显示单元数组 A 的全部内容
A{1,1} =
      8     1     6
      3     5     7
      4     9     2
A{2,1} =
      1     2     3     4     5
A{1,2} =
10/1/2004
```

```
A{2,2} =

    1.0000 + 2.0000i        2.0000 - 3.0000i
    2.0000 + 3.0000i        1.0000 - 2.0000i
```

(2) 单元索引。将单元数组的单元号用花括号括起来，在等式右侧写入单元的内容。同上例，书写如下：

```
>>A{1,1} = magic(3);
>>A{1,2} = '10/1/2004';
>>A{2,1} = 1:5;
>>A{2,2} = [1 + 2* i  2 - 3* i;2 + 3* i  1 - 2* i];
>>celldisp(A)
A{1,1} =

     8     1     6
     3     5     7
     4     9     2

A{2,1} =

     1     2     3     4     5

A{1,2} =
10/1/2004
A{2,2} =

    1.0000 + 2.0000i        2.0000 - 3.0000i
    2.0000 + 3.0000i        1.0000 - 2.0000i
```

对于单元数组中，单元中的单元数组称为嵌套的单元数组或称为单元数组的子集。对于子集的赋值可以通过在两次单元索引（两次索引仍用花括号）的等式右侧赋值。详见如下程序。

上述表 3-1 可以用以下语句赋值：

```
>>C = cell(2,3)                                   %预设单元数组的大小
C =

    []     []     []
    []     []     []

>>C{1,1} = magic(3);                              % 设置（1，1）单元的内容，要用花括号
>>C{1,2} = strvcat('John Smith','3/8/2005','Class II');  %设置（1，2）单元的内容
>>C{1,3} = [2 + i 1 - 2* i;1 + 2* i 2 - i];       %设置（1，3）单元的内容
>>C{2,1} = [1.4  2.8  3.3  5.6];                  %设置（2，1）单元的内容
>>C{2,2} = [2 3 4;3 6 10;4 10 20];                %设置（2，2）单元的内容
>>C{2,3}{1,1} = 'text';                           %设置（2，3）单元的子集（1，1）内容
```

```
>>C{2,3}{1,2} = [4 2;1 5];                %设置（2，3）单元的子集（1，2）内容
>>C{2,3}{2,1} = [4 2 7];                  %设置（2，3）单元的子集（2，1）内容
>>C{2,3}{2,2} = [0.5 + 6* i];             %设置（2，3）单元的子集（2，2）内容
>>C                                       %显示单元数组的结构
C =
    [3x3 double]      [3x10 char ]      [2x2 double]
    [1x4 double]      [3x3 double]      {2x2 cell }
>>celldisp(C)                             %显示单元数组 C 的全部内容，若celldisp(C)改
                                            写成celldisp(C,name)，则以 name 代替 C 来
                                            显示单元数组
C{1,1} =
     8     1     6
     3     5     7
     4     9     2
C{2,1} =
    1.4000    2.8000    3.3000    5.6000
C{1,2} =
John Smith
3/8/2005
Class II
C{2,2} =
     2     3     4
     3     6    10
     4    10    20
C{1,3} =
   2.0000 + 1.0000i   1.0000 - 2.0000i
   1.0000 + 2.0000i   2.0000 - 1.0000i
C{2,3}{1,1} =
text
C{2,3}{2,1} =
     4     2     7
C{2,3}{1,2} =
     4     2
     1     5
C{2,3}{2,2} =
   0.5000 + 6.0000i
```

```
≫cellplot(C)                       %用图形显示单元数组 C 的结构图如图 3-9 所示，矩形代表向量元素或矩阵元素，标量或短的文本字符串则表示文本
```

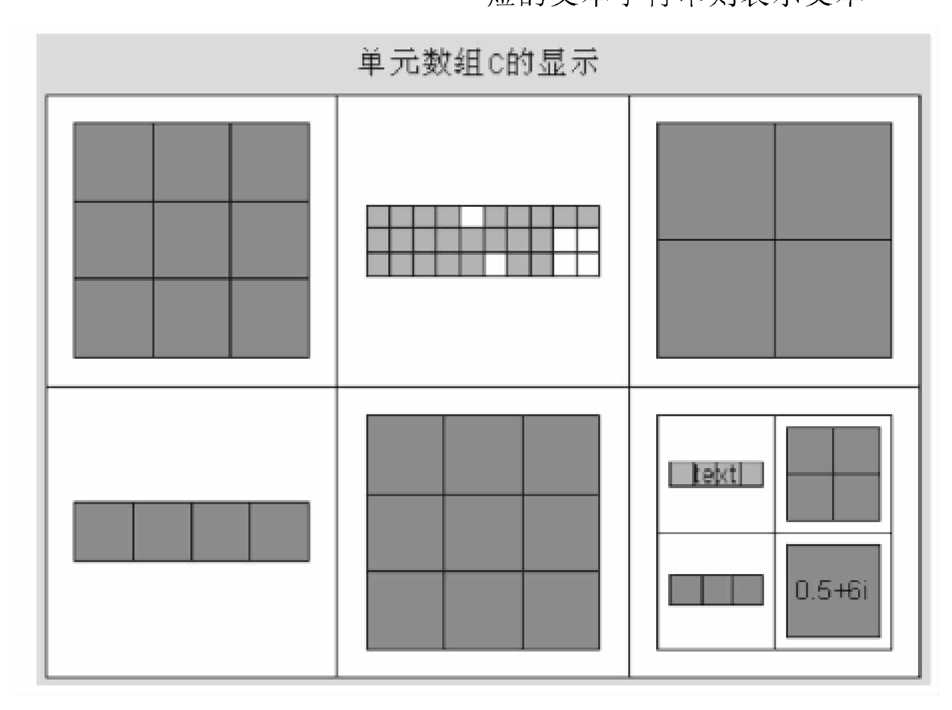


图 3-9　单元数组的 C 的图形显示

3.16.2　单元数组的删除和改写

与一般数组相似，可以用赋值空矩阵来删除单元数组的某单元。亦可以用 reshape 来改写矩阵。现举例如下：

【例 3-23】　已知单元数组A = {magic(3),'10/1/2004','name _ wang';[1 3 5 7],eye(3),'$5000'}，要求删除单元{1,1}，{2,1}和{2,2}。

解：

```
≫A = {magic(3),'10/1/2005','name _ Wang';[1 3 5 7],eye(3),'$5000'}
A =                                         %输入单元数组 A
    [3x3 double]     '10/1/2004'     'name _ Wang'
    [1x4 double]     [3x3 double]    '$5000'
≫A{1,1} = []                                %删除 (1, 1) 单元
A =
    []               '10/1/2005'     'name _ Wang'
    [1x4 double]     [3x3 double]    '$5000'
≫A{2,1} = []                                %删除 (2, 1) 单元
A =
    []               '10/1/2004'     'name _ Wang'
    []               [3x3 double]    '$5000'
```

```
>>A{2,2} = []                          %删除（2，2）单元
A =
    []              '10/1/2004'        'name_Wang'
    []              []                 '$5000'
```

【例 3-24】 已知单元数组 A = cell(4,6)，要求改写成为 3 行 8 列的单元数组 B，再取 B 矩阵的第 4～8 列组成单元数组 C。

```
>>A = cell(4,6)                        %预分配单元数组 A
A =
    []    []    []    []    []    []
    []    []    []    []    []    []
    []    []    []    []    []    []
    []    []    []    []    []    []
>>B = reshape(A,3,8)                   %改写成 3 行 8 列
B =
    []    []    []    []    []    []    []    []
    []    []    []    []    []    []    []    []
    []    []    []    []    []    []    []    []
>>C = B(:,4:8)                         %取 B 的 4～8 列得单元数组 C
C =
    []    []    []    []    []
    []    []    []    []    []
    []    []    []    []    []
```

3.16.3 单元数组的运算

单元数组中的数值矩阵仍符合数值矩阵的运算规则，对于字符数组则仍符合字符数组的运算规则。下面举例予以说明。

【例 3-25】 已知单元数组 A {1,1} = rand(2,3)，A {2,1} = [1 3 5 7]，A {1,2} = rand(3,2)，A {2,2} = 1:4，求单元元素的乘积 A{1,1}* A{1,2}并存入 A{1,3}，以及单元元素的卷积（详见第 6 章）conv(A{2,1},A{2,2})并存入 A{2,3}。

```
>>A = cell(2,3)                        %对单元数组 A 预分配
A =
    []    []    []
    []    []    []
>>A{1,1} = rand(2,3);                  %2 行 3 列的随机矩阵赋入A{1,1}
>>A{1,2} = rand(3,2);                  %3 行 2 列的随机矩阵赋入A{1,2}
>>A{2,1} = [1 3 5 7];                  %向量赋入A{2,1}
>>A{2,2} = 1:4;                        %向量赋入A{2,2}
```

```
>>A{1,3}=A{1,1}*A{1,2};                    %乘积赋入A{1,3}
>>A{2,3}=conv(A{2,1},A{2,2})               %卷积赋入A{2,3}
A =
    [2x3 double]      [3x2 double]      [2x2 double]
    [1x4 double]      [1x4 double]      [1x7 double]
>>celldisp(A)                              %显示单元数组 A
A{1,1} =
    0.9218    0.1763    0.9355
    0.7382    0.4057    0.9169
A{2,1} =
     1     3     5     7
A{1,2} =
    0.4103    0.3529
    0.8936    0.8132
    0.0579    0.0099
A{2,2} =
     1     2     3     4
A{1,3} =                                   %矩阵乘积
    0.5899    0.4778
    0.7185    0.5994
A{2,3} =                                   %向量卷积
     1     5    14    30    41    41    28
```

3.17 结构数组

结构数组是对某个对象，不同属性的数据，用一组数据来表示，称为结构数组。结构数组中的元素可以是数值，亦可以是字符串。因此产品档案、库存档案、病历档案、人事档案、检测数据等都是结构数组的例子。MATLAB中有关结构数组的函数列举在表3-2中。

表 3-2 有关结构数据的函数

函 数 名	说 明
struct	创建结构数组
fieldname	提取结构数组字段名
getfield	提取字段名内容
isstruct	是结构数组返回 1
isfield	是结构数组的字段名则返回 1
rmfield	删除字段名
struct2cell	将结构数组转换成单元数组
Cell2struct	将单元数组转换成结构数组

3.17.1 结构数组的创建

结构数组的创建，可以采取直接赋值或用函数 struct 来创建。

【例 3-26】 表 3-3 是几个学生的档案。数组名为 student，试用结构数组来表示。

表 3-3 学生档案

ID	Name	Age	Sex	score
101	wang	21	F	88
102	Chang	20	M	95
103	Li	19	F	79
104	Lu	23	M	73

解：

用直接赋值学生的标识 ID 为

```
>>student(1).ID = 101;        %student 为数组名，ID 为字段名，中间必须用句号隔开
>>student(2).ID = 102;
>>student(3).ID = 103;
>>student(4).ID = 104;
```

用直接赋值学生的姓名 name 为

```
>>student(1).name = 'wang';
>>student(2).name = 'Chang';
>>student(3).name = 'Li';
>>student(4).name = 'Lu';
```

用直接赋值学生的年龄 age 为

```
>>student(1).age = 21;
>>student(2).age = 20;
>>student(3).age = 19;
>>student(4).age = 23;
```

用直接赋值学生的性别 sex 为

```
>>student(1).sex = 'F';
>>student(2).sex = 'M';
>>student(3).sex = 'F';
>>student(4).sex = 'M';
```

用直接赋值学生的成绩 score 为

```
>>student(1).score = 88;
>>student(2).score = 95;
>>student(3).score = 79;
>>student(4).score = 73;
```

至此结构数组输入完成。当键入struct(student)则显示为

```
>>student
```

```
student =
1x4 struct array with fields:                    %显示有4个数组，字段名为ID，name，age，sex和score
    ID
    name
    age
    sex
    score
```

如果用结构数组函数 struct 创建，则书写格式为

$$array_name = struct('field1', val1, 'field2', val2, \cdots) \tag{3-13}$$

式中，array _ name——结构数组名，field1，field2——为字段名，val1，val2——为对应字段名的值。

3.17.2 结构数组与单元数组的转换

结构数组与单元数组的转换函数为

$$c = struct2cell(s) \tag{3-14}$$

也可以进行逆转换，逆转换函数为

$$s = cell2struct(c) \tag{3-15}$$

式中，s 为结构数组名，c 为单元数组名。

假若结构数组 s 是多维的，则转换后的单元数组的大小为

$$[p, size(s)] \tag{3-16}$$

式中，p 为结构数组 s 的字段数。

【例 3-27】 今有树的品种为白桦，树高为 28.5m 的结构数组信息，将其转换成单元素组。

解：

```
>>S = struct('category','tree','heigh',28.5,'name','brich')    %创建结构数组
S =
    category: 'tree'                                           %字段，category
        height: 28.5000                                        %字段，height
          name: 'birch'                                        %字段，name
>>C = struct2cell(S)                                           %转换成单元数组
C =
    'tree'
    [28.5000]
    'birch'
>>size(C)                                                      %转换后的单元数组大小
ans =
    3     1
```

【例 3-28】 顺接【例 3-26】，将结构数组 student 转换成单元数组 C。

解：

```
>>student                                          %显示结构数组的结构
student =
1x4 struct array with fields:
    ID
    name
    sex
    score
>>C = struct2cell(student);                        %转换结构数组 student 成单元数组 C
>>C = [C(:,1),C(:,2),C(:,3),C(:,4)]               %显示单元数组
C =
 [101]         [102]         [103]         [104]
 'wang'        'Chang'       'Li'          'Lu'
 'F'           'M'           'F'           'M'
 [88]          [95]          [79]          [73]
```

3.17.3 单元数组的数据处理

MATLAB 的单元数组虽然有类似于 Excel 的功能，但其统计分析的功能不如 Excel。例如求【例 3-26】考分平均数，不能直接使用平均值函数 mean，当使用时，命令窗口会显示出错，并提示函数 mean 不适用单元数组。要计算上例的平均值，必须用以下方法。将单元数组中单元的数值转换成数值向量，随后进行数值运算。现举例如下。

【例 3-29】 求【例 3-26】中单元数组 C 的考分 score 的平均值。

解：

在命令窗口输入如下程序：

```
>>s = 0;                                           %设初值
>>for i = 1:4                                      %取 i 等于 1:4 的循环
s(i) = C{4,i};                                     %将数组 C 的第四行第 i 列的内容赋
                                                     入数值数组 s
end,s,mean(s)                                      %显示 s 及 s 的平均值
s =                                                %考分成绩向量
     88      95      79      73
ans =                                              %考分成绩平均值
    83.7500
```

3.18 多维数组

众所周知，矩阵是用得最广的二维数组。但是现实世界上有很多事物是多于二维的。物理学上的波以尔定律，就是以气体压强、气体容积和绝对温度的三维数据来描述的。库存物

资的安放位置，是以库房号、货架号、层号和列号的四维数据来定位。人事档案，则是以姓名、个人属性和个人属性值的三维数据来描述。多维数组的一般表达式为

$$A\ (i,\ j,\ k\cdots)$$

式中，A 为多维数组名；i、j、k…为一维、二维、三维…下标索引值。通常 i 称行，j 称列，k 称页。

3.18.1 多维数组的创建

(1) 多维数组的创建可以通过直接赋值来取得。

【例 3-30】 设置三维矩阵 **A**，第 1 页为magic(3)，第 2 页为vander([1,2,3])，第 3 页为pascal(3)。

解:

```
>>A(:,:,1) = magic(3)                        %第 1 页，设为magic(3)
A =

     8     1     6
     3     5     7
     4     9     2

>>A(:,:,2) = vander([1 2 3])                 %第 2 页，设为范德蒙矩阵
A(:,:,1) =

     8     1     6
     3     5     7
     4     9     2

A(:,:,2) =

     1     1     1
     4     2     1
     9     3     1

>>A(:,:,3) = pascal(3)                       %第 3 页，设为 pascal 矩阵
A(:,:,1) =

     8     1     6
     3     5     7
     4     9     2

A(:,:,2) =

     1     1     1
     4     2     1
     9     3     1

A(:,:,3) =

     1     1     1
     1     2     3
```

```
    1     3     6
```

(2) 通过矩阵函数生成多维函数

【例 3-31】 设置三维正态分布的随机矩阵 $\boldsymbol{B}$，其中 $m=n=4$，$k=3$。

解：

```
≫B = rand(4,4,3)
B(:,:,1) =                              %随机矩阵第 1 页

    0.9501    0.8913    0.8214    0.9218
    0.2311    0.7621    0.4447    0.7382
    0.6068    0.4565    0.6154    0.1763
    0.4860    0.0185    0.7919    0.4057
B(:,:,2) =                              %随机矩阵第 2 页

    0.9355    0.0579    0.1389    0.2722
    0.9169    0.3529    0.2028    0.1988
    0.4103    0.8132    0.1987    0.0153
    0.8936    0.0099    0.6038    0.7468
B(:,:,3) =                              %随机矩阵第 3 页

    0.4451    0.8462    0.8381    0.8318
    0.9318    0.5252    0.0196    0.5028
    0.4660    0.2026    0.6813    0.7095
    0.4186    0.6721    0.3795    0.4289
```

【例 3-32】 用矩阵复制生成多维矩阵 $\boldsymbol{C}$，它是由矩阵 $\boldsymbol{A}=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$，复制成 $3\times4\times3$ 矩阵。

解：

```
≫A = [1 2;3 4]                          %样板矩阵
A =

    1     2
    3     4

≫C = repmat(A,[3,4,3])                  %将矩阵 A 复制成 3*4，复制后再复制 3 页
C(:,:,1) =

    1     2     1     2     1     2     1     2
    3     4     3     4     3     4     3     4
    1     2     1     2     1     2     1     2
    3     4     3     4     3     4     3     4
    1     2     1     2     1     2     1     2
    3     4     3     4     3     4     3     4

C(:,:,2) =
```

```
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
C(:,:,3) =
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
     1     2     1     2     1     2     1     2
     3     4     3     4     3     4     3     4
```

【例 3-33】 用矩阵连接函数 cat，生成多维数组D = cat(dim, A, B, C), A = magic(3), B = ones(3), C = eye(3)，维数 dim = 3。

解:

```
>>A = magic(3)                          %输入矩阵 A
A =
     8     1     6
     3     5     7
     4     9     2
>>B = ones(3)                           %输入矩阵 B
B =
     1     1     1
     1     1     1
     1     1     1
>>C = eye(3)                            %输入矩阵 C
C =
     1     0     0
     0     1     0
     0     0     1
>>D = cat(3, A, B, C)                   %沿第 3 维连接矩阵 A、B、C，即成三维数组 D
D(:,:,1) =
     8     1     6
     3     5     7
     4     9     2
D(:,:,2) =
```

```
     1     1     1
     1     1     1
     1     1     1
D(:,:,3) =
     1     0     0
     0     1     0
     0     0     1
```

3.18.2 多维数组的运算

多维数组的运算与二维数组的运算基本相同。MATLAB的大多数内装函数支持多维数组，例如sum、mean、size、ones、zeros、eye、rand、randn和prod等。在某些函数不支持多维数组时，可以分别用二维数组经多次运算来代替。

```
>>B(:,:,1) = magic(3);                %第1页为3阶魔方矩阵
>>B(:,:,2) = ones(3);                 %第2页为3阶全1矩阵
>>B(:,:,3) = eye(3);                  %第3页为3阶幺矩阵
>>size(B)                             %B矩阵的大小，为3行、3列、3页
ans =
     3     3     3
>>sum(B)                              %数组求和
ans(:,:,1) =
    15    15    15
ans(:,:,2) =
     3     3     3
ans(:,:,3) =
     1     1     1
>>prod(B)                             %数组求乘积
ans(:,:,1) =
    96    45    84
ans(:,:,2) =
     1     1     1
ans(:,:,3) =
     0     0     0
```

3.18.3 猜数游戏

作为多维数组的一个例子，今举一个猜数游戏。游戏一开始，提出7张数据表。请你查看，你心中默想的数字在那几张表中有你默想的数字（默想的数字限制在1~127，如果想

扩大猜数的范围，须增加表数）。随后依程序提示，输入向量。若某个表中有默想数字，则输入向量中的相应元素置 1，否则置零。程序如下：

```
% This program for guess a number, which is less then integer of 127
clear                                    %清内存
i1 = 1:2:127;
A(:,:,1) = reshape(i1,8,8);              %设计第一表，奇数，表中数字转换成二进制数
                                           的个位均为 1，reshape 为矩阵重新排列函数
i2 = 1:2:63;
A2 = [2*i2;2*i2+1];
A(:,:,2) = reshape(A2,8,8);              %设计第二表，二进制数的第 2 位均为 1
i4 = 1:2:31;
A3 = [4*i4;4*i4+1;4*i4+2;4*i4+3];
A(:,:,3) = reshape(A3,8,8);              %设计第三表，二进制数的第 3 位均为 1
i8 = 1:2:15
A(:,:,4) = [8*i8;8*i8+1;8*i8+2;8*i8+3;8*i8+4;8*i8+5;8*i8+6;8*i8+7];
                                         %设计第四表，二进制数的第 4 位均为 1
A5 = [16:31,48:63,80:95,112:127]';
A(:,:,5) = reshape(A5,8,8);              %设计第五表，二进制数第 5 位均为 1
A6 = [32:63,96:127]';
A(:,:,6) = reshape(A6,8,8);              %设计第六表，二进制数第 6 位均为 1
A7 = [64:127]';
A(:,:,7) = reshape(A7,8,8);              %设计第七表，二进制数第 7 位均为 1
A                                        %显示 1~7 张数表
V = input('please input a vector as [0,0,1,0,1,1,0],which matrix has your number set 1 else
set 0 \n')                               %输入向量，对含有默想数字的表，在向量中将
                                          对应元素置 1，否则置零
S = sum(V(1)+2*V(2)+4*V(3)+8*V(4)+16*V(5)+32*V(6)+64*V(7));
fprintf('your number is %d \n',S)        %输出猜想数
```

运行猜数游戏 guessnum 如下：

```
>>guessnum                               %调用猜数游戏，显示 7 张数表
A(:,:,1) =

     1    17    33    49    65    81    97   113
     3    19    35    51    67    83    99   115
     5    21    37    53    69    85   101   117
     7    23    39    55    71    87   103   119
     9    25    41    57    73    89   105   121
    11    27    43    59    75    91   107   123
    13    29    45    61    77    93   109   125
```

```
    15    31    47    63    79    95   111   127

A(:,:,2) =

     2    18    34    50    66    82    98   114
     3    19    35    51    67    83    99   115
     6    22    38    54    70    86   102   118
     7    23    39    55    71    87   103   119
    10    26    42    58    74    90   106   122
    11    27    43    59    75    91   107   123
    14    30    46    62    78    94   110   126
    15    31    47    63    79    95   111   127

A(:,:,3) =

     4    20    36    52    68    84   100   116
     5    21    37    53    69    85   101   117
     6    22    38    54    70    86   102   118
     7    23    39    55    71    87   103   119
    12    28    44    60    76    92   108   124
    13    29    45    61    77    93   109   125
    14    30    46    62    78    94   110   126
    15    31    47    63    79    95   111   127

A(:,:,4) =

     8    24    40    56    72    88   104   120
     9    25    41    57    73    89   105   121
    10    26    42    58    74    90   106   122
    11    27    43    59    75    91   107   123
    12    28    44    60    76    92   108   124
    13    29    45    61    77    93   109   125
    14    30    46    62    78    94   110   126
    15    31    47    63    79    95   111   127

A(:,:,5) =

    16    24    48    56    80    88   112   120
    17    25    49    57    81    89   113   121
    18    26    50    58    82    90   114   122
    19    27    51    59    83    91   115   123
    20    28    52    60    84    92   116   124
    21    29    53    61    85    93   117   125
    22    30    54    62    86    94   118   126
```

```
    23    31    55    63    87    95   119   127
A(:,:,6) =
    32    40    48    56    96   104   112   120
    33    41    49    57    97   105   113   121
    34    42    50    58    98   106   114   122
    35    43    51    59    99   107   115   123
    36    44    52    60   100   108   116   124
    37    45    53    61   101   109   117   125
    38    46    54    62   102   110   118   126
    39    47    55    63   103   111   119   127
A(:,:,7) =
    64    72    80    88    96   104   112   120
    65    73    81    89    97   105   113   121
    66    74    82    90    98   106   114   122
    67    75    83    91    99   107   115   123
    68    76    84    92   100   108   116   124
    69    77    85    93   101   109   117   125
    70    78    86    94   102   110   118   126
    71    79    87    95   103   111   119   127
please input a vector as [0,0,1,0,1,1,0],which matrix table has your number set 1 on corresponding element else set 0    %输入向量，指出那几张表中有默想数字
[1 1 0 0 0 1 1]                         %表 1，2 ，6，7，含有默想数字
V =
     1     1     0     0     0     1     1
your number is 99                       %此数为 99
```

3.18.4　15 个滑块游戏

15 个滑块游戏，它是作为学习 GUI（图形用户界面）的演示程序，也可以作为在操作 MATLAB 中的休闲娱乐用。

游戏的开始，是通过在命令窗口键入 fifteen 而进入程序，并出现如图 3-10 所示的画面。

fifteen 程序是一个由 15 个数字方块组成的，存放在 4×4 = 16 个位置可滑动的矩形拼图中。胜利者是设法通过移动数字方块，使 15 个方块按行顺序排列，留下一个空位置在右下角。用鼠标点击在行或列处有缺口的方块，那么被点击的方块就移向空位。游戏者应设法用最少的步数，使 15 个方块达到顺序排列。游戏者可以点击“Help”按钮得到帮助。点击“New Game”按钮重新开始。点击“Close”结束程序。

Fifteen 的程序是可读的。它的程序是处于目录 matlab/toolbox/demos/fifteen。当键入 type fifteen 即可查看程序的内容。但版本不同，则所处的目录可能不同。

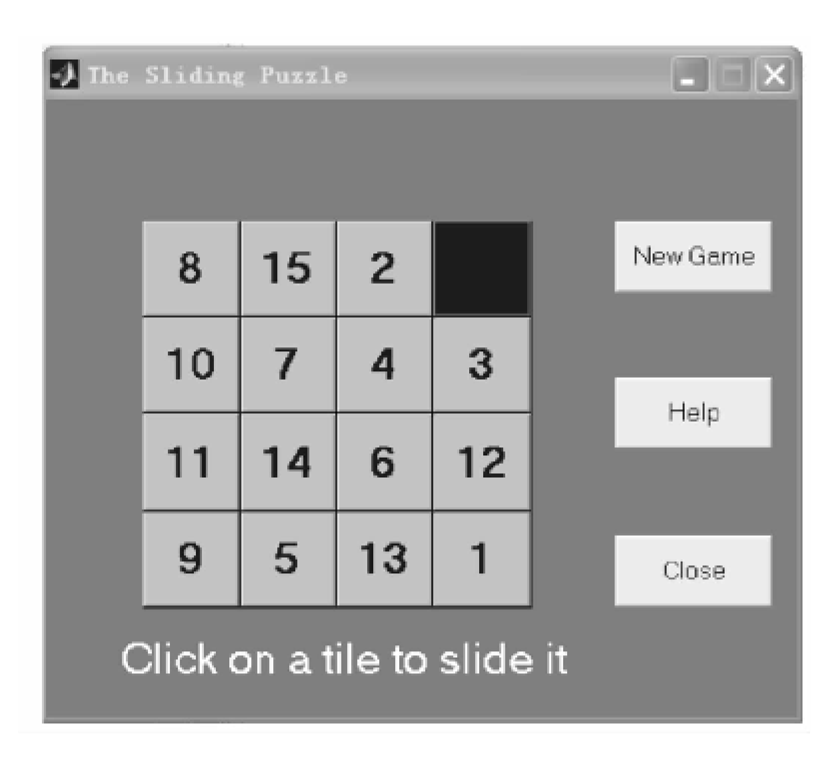


图 3-10　15 个滑块游戏

第 3 章习题

3-1　用内联函数编写函数 $y=\dfrac{\sin x}{1+3x+x^2}$。

3-2　用函数 M 文件编写函数 $Z=x_1{}^2+2x_2{}^2+3x^2+2x_1x_2+3x_2x_3+5x_2x_3$。

3-3　编制一程序，计算某正整数的逆序数程序，并计算两个数 84723536、52397899 的逆序数。

3-4　编制一程序寻找 m 行 n 列的矩阵中的最大元素和最小元素，及其所在的相应位置的下标。

3-5　编制一程序，在已知向量 V 中寻找中位数（提示：参考平均值函数 mean）。

3-6　编制一程序，当输入一正整数 n，创建一个顺序数 1：n^2 组成的方阵，方阵排列的次序是顺序的，如 $\begin{vmatrix}1&2&3\\4&5&6\\7&8&9\end{vmatrix}$ 所示。

3-7　已知 9 阶稀疏矩阵 $\boldsymbol{A}=\begin{vmatrix}0&0&0&0&1&0&0&0&0\\0&0&0&6&0&2&0&0&0\\0&0&11&0&7&0&3&0&0\\0&16&0&12&0&8&0&4&0\\21&0&17&0&13&0&9&0&5\\0&22&0&18&0&14&0&10&0\\0&0&23&0&19&0&15&0&0\\0&0&0&24&0&20&0&0&0\\0&0&0&0&25&0&0&0&0\end{vmatrix}$，请设法通过行、列叠加后提取其中 5 阶魔方矩阵。

3-8　编制一程序，列出 1100 内的素数。

3-9　编制一程序，已知同维向量 $\boldsymbol{X}$、$\boldsymbol{Y}$，求其相关系数 $\boldsymbol{R}_{xy}$。

$$R_{xy}=\frac{S_{xy}}{\sqrt{S_{xx}S_{yy}}},\ S_{xx}=\sum_{i=1}^{n}(x_i-\bar{x})^2,\ S_{yy}=\sum_{i=1}^{n}(y_i-\bar{y})^2,\ S_{xy}=\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})$$

第 4 章　线性方程组的数值解和代数方程组的符号解

线性方程组在线性代数中占有重要的地位。高斯消去法、迭代解法、矩阵的秩、矩阵的条件数、矩阵的特征值、矩阵的范数，都是从线性方程组的求解发展起来的。其次电气工程中的线性电路分析，自动控制系统中的状态变量求解，静力学和结构力学中的力的分析都离不开线性方程组的求解。

线性方程组的一般形式为

$$\boldsymbol{AX}=\boldsymbol{b} \tag{4-1}$$

式中，系数矩阵 $\boldsymbol{A}$ 为 m 行 n 列，$\boldsymbol{X}$ 为 n 行的列向量，$\boldsymbol{b}$ 为 m 行的列向量。

线性方程组（4-1）的求解，可以根据行、列的数值不同，分为 3 种情况：

（1）确定方程组——矩阵 $\boldsymbol{A}$ 的行数 m 与列数 n 相等，且 $\det\boldsymbol{A}$ 不等于零，则方程组有惟一解。

（2）超定方程组——矩阵 $\boldsymbol{A}$ 的行数 m，大于列数 n，则方程组无精确解，但存在近似的最小二乘解。

（3）不定方程组——矩阵 $\boldsymbol{A}$ 的行数 m，小于列数 n，则方程组有无穷多个解。若存在有约束条件，则仍有解，但这属于线性规划的范围。

4.1　确定方程组

确定方程组的求解可以采用高斯消去法或迭代解法。但在 MATLAB 中，可以使用矩阵左除法（即使用反斜杠符号）或者通过逆矩阵函数 inv 来求解。

若确定方程组为 $\boldsymbol{AX}=\boldsymbol{b}$，其中 $\boldsymbol{A}$ 为方阵，则其解为

$$X = A \setminus b \tag{4-2}$$

或

$$X = inv(A) * b \tag{4-3}$$

下面举例说明它的应用。

【例 4-1】　有一数组 x_1，x_2，…，x_5，其总和为 898，$x_3=x_1+x_2$，$x_4=x_2+x_3$，$x_5=x_3+x_4$，且 x_5 与 x_1 之差为 322，求解此数组。

解：

根据题意列出下列线性方程组：

$$x_1+x_2+x_3+x_4+x_5=898 \tag{4-4}$$

$$x_1+x_2-x_3=0 \tag{4-5}$$

$$x_2+x_3-x_4=0 \tag{4-6}$$

$$x_3+x_4-x_5=0 \tag{4-7}$$

$$-x_1+x_5=322 \tag{4-8}$$

设待求数组为列向量 $\boldsymbol{X}$，其元素为 x_1，x_2，…，x_5，根据上述方程组可得系数矩阵 $\boldsymbol{A}$，在 MATLAB 命令窗口输入如下程序：

```
≫A = [1 1 1 1 1;1 1 -1 0 0;0 1 1 -1 0;0 0 1 1 -1; -1 0 0 0 1]   %输入线性方程组系数矩阵 A
A =
     1     1     1     1     1
     1     1    -1     0     0
     0     1     1    -1     0
     0     0     1     1    -1
    -1     0     0     0     1
≫b = [898 0 0 0 322]'              % 输入线性方程组右侧列向量
b =
   898
     0
     0
     0
   322
≫X = inv(A)* B                     %用逆矩阵法解线性方程组
X =
   55                              %即 x1
   89                              %即 x2
  144                              %即 x3
  233                              %即 x4
  377                              %即 x5
≫X = A \ b                         %用矩阵左除法解线性方程组,结果是一致的
   X =
       55
       89
      144
      233
      377
```

【例 4-2】 求下列桥式电路中的支路电流 i_1, i_2, i_3 的符号表达式,并且当 $E = 12V$, $r_1 = 50\Omega$, $r_2 = 100\Omega$, $r_3 = 20\Omega$, $r_4 = 40\Omega$, $r_5 = 100\Omega$ 时的支路电流。

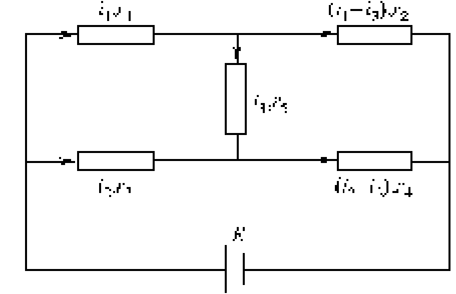


图 4-1 桥式电路

解:

由回路电流法可得下列方程式:

$$i_1 r_1 + (i_1 - i_3) r_2 = E \tag{4-9}$$

$$i_1 r_1 - i_2 r_3 + i_3 r_5 = 0 \tag{4-10}$$

$$(i_1 - i_3) r_2 - i_3 r_5 - (i_2 + i_3) r_4 = 0 \tag{4-11}$$

在 MATLAB 命令窗口输入如下程序:

```
≫syms r1 r2 r3 r4 r5 E real                                    %设置符号变量,申明变量是实数
≫A = [(r1 + r2),0, - r2;r1, - r3,r5;r2, - r4, - (r2 + r4 + r5)]
                                                               %输入回路方程式(4-9)~式(4-11)的系数矩阵
  A  =                                                         %矩阵A的符号表达形式
  [      r1 + r2,          0,              - r2]
  [           r1,       - r3,                r5]
  [           r2,       - r4,   - r2 - r4 - r5]
  ≫b = [E,0,0]'                                                %输入向量b
  b  =
  [E]
  [0]
  [0]
  ≫ I = inv(A)* B                                              %解支路电流向量I的符号表达式,
  I =                                                          %I = [i1;i2;i3]
  [(r3* r2 + r3* r4 + r3* r5 + r5* r4)/(r1* r3* r2 + r1* r3* r4 + r1* r3* r5 + r1* r5* r4 + r2*
r3* r4 + r2* r3* r5 + r2* r5* r4 + r1* r2* r4)* E]
  [(r1* r2 + r1* r4 + r1* r5 + r5* r2)/ (r1* r3* r2 + r1* r3* r4 + r1* r3* r5 + r1* r5* r4 + r2*
r3* r4 + r2* r3* r5 + r2* r5* r4 + r1* r2* r4)* E]
  [- (r1* r4 - r3* r2)             / (r1* r3* r2 + r1* r3* r4 + r1* r3* r5 + r1* r5* r4 + r2*
r3* r4 + r2* r3* r5 + r2* r5* r4 + r1* r2* r4)* E]

  ≫ r1 = 50;r2 = 100;r3 = 20;r4 = 40;r5 = 100;E = 12;           %以桥式电路参数代入上述符号变量,使符号变量转换成数值
  ≫A = [(r1 + r2),0, - r2;r1, - r3,r5;r2, - r4, - (r2 + r4 + r5)] %将符号矩阵转换成数值矩阵,为避免重复输入,可在命令历史窗口双击已经输入过的矩阵A,则命令窗口出现数值矩阵A

  A =
  150      0   - 100
   50   - 20     100
  100   - 40   - 240
  ≫b =  [12 0 0]';                                              % 输入电路线性方程组等式右侧列向量
```

```
≫I = inv (A)* B                                   %解支路电流矩阵
I  =
     0.0800                                       %i1 支路电流，单位为安培
     0.2000                                       %i2 支路电流，单位为安培
          0                                       %i3 支路电流，单位为安培，
                                                    由于电桥平衡，所以 i3 支
                                                    路无电流
```

【例 4-3】 某工厂生产 A、B 和 C 三种产品，每项产品都经过机械加工、电镀和总装。每项产品的加工工时和工厂的生产能力见表 4-1。如果充分利用工厂生产能力，问每月能生产 A、B 和 C 产品各多少件。

表 4-1　每项产品的加工工时和工厂生产能力

加工工时 \ 品种 / 工序	A	B	C	每月可利用工时
机械加工	1.0	1.1	1.4	4450
电镀	0.5	0.55	0.55	2200
装配	0.55	0.75	0.85	2600

解：

设 A 产品生产数量为 $X(1)$，B 产品生产数量为 $X(2)$，C 产品生产数量为 $X(3)$，则

$$X(1)+1.1X(2)+1.4X(3)=4450 \tag{4-12}$$

$$0.5X(1)+0.55X(2)+0.55X(3)=2200 \tag{4-13}$$

$$0.55X(1)+0.75X(2)+0.85X(3)=2600 \tag{4-14}$$

由此得系数矩阵为

```
≫A = [1 1.1 1.4;0.5 0.55 0.55;0.55 0.75 0.85]
A  =
     1.0000     1.1000     1.4000
     0.5000     0.5500     0.5500
     0.5500     0.7500     0.8500
≫b = [4450 2150 2600]';             %等式右侧向量
≫X = inv(A)* b                       %所求品种向量
X  =
   1.0e+003*
     2.8966
     0.7759
     0.5000
≫ fix(X(1))                          %取整后的 A 产品数,fix 为取整函数
ans  =
     2896
≫ fix(X(2))                          %取整后的 B 产品数
```

```
ans  =
      775
≫ fix(X(3))                        %取整后的 C 产品数
ans  =
      500
```

4.2 超定方程组

超定方程组在数学上称为矛盾方程组。例如有两个工件 A 和 B，A 的实测长度为 5.01cm，B 的实测长度为 4.98cm，把 A、B 两个产品叠加起来的实测长度为 10.02cm。这在数学上看起来是矛盾的。因为根据计算 A + B = 9.99，怎么会变成 10.02 呢。但在实践上是可能的，因为测量时总有误差。这个例子中，它有 3 个方程式，两个未知数。这是最简单超定方程组的例子。超定方程组在实践上的例子是很多的，如大地测量，统计分析与预测，超静定结构分析，试验数据的曲线拟合等。下面将举例予以说明。

超定方程组的解法在采用了 MATLAB 以后，变得很方便。它不需要检查系数矩阵的秩数（即 rank）是否小于行数、列数，用广义逆矩阵（即 pinv）进行计算即可。但是计算结果是满足最小二乘解。即使计算误差［b - A* X］的 2 范数最小。超定方程组的计算公式为

$$X = pinv(A) * b \tag{4-15}$$

式中，A 为超定方程组的系数矩阵，b 为方程组等式右侧的列向量。下面举例说明超定方程组的应用。

【例 4-4】 大地测量中，测得平面三角形的 3 内角分别为 60.5 度，71.4 度和 48.5 度，求最小二乘解，并修正测量值。

解：

设 3 个角的修正值分别为 X_1，X_2 和 X_3，则

$$X_1 = 60.5 \tag{4-16}$$

$$X_2 = 71.4 \tag{4-17}$$

$$X_3 = 48.5 \tag{4-18}$$

$$X_1 + X_2 + X_3 = 180 \tag{4-19}$$

式（4-16）～式（4-18）为实测数据，式（4-19）为平面上，三角形中三内角之和为 180 度。这是一组有 3 个变量，4 个方程式的超定方程组。在 MATLAB 命令窗口输入如下程序：

```
≫A = [1 0 0;0 1 0;0 0 1;1 1 1]          %输入本例方程组的系数矩阵
A  =
     1     0     0
     0     1     0
     0     0     1
     1     1     1
≫b = [60.5 71.4 48.5 180]'              %输入列向量 b
b =
```

```
    60.5000
    71.4000
    48.5000
   180.0000
>>X = pinv(A) * b                       %解超定方程组
X  =
    60.4000
    71.3000
    48.4000
>>b - A * X                             %计算误差向量
ans =
    0.1000
    0.1000
    0.1000
   -0.1000
>>norm(ans)                             %误差向量的 2 范数
ans =
    0.2000
```

根据上述计算，角度的修正值应为 60.4，71.3 和 48.4。三角形 3 内角之和为 180.1。

【例 4-5】 今有一组试验数据，时间向量 $\boldsymbol{t}$ = [0 0.3 0.8 1.1 1.6 2.3]，输出向量 $\boldsymbol{y}$ = [0.82 0.72 0.63 0.6 0.55 0.5]，考虑用衰减指数曲线拟合它。求衰减指数曲线的系数，绘出散点图和拟合曲线。

解：

设衰减的指数曲线方程为

$$\boldsymbol{y} = c_1 + c_2 \exp(-\boldsymbol{t}) \tag{4-20}$$

将时间向量 $\boldsymbol{t}$ 代入，得一组 6 个方程式的超定方程组即：

$$c_1 + c_2 = 0.82 \tag{4-21}$$

$$c_1 + 0.7408 c_2 = 0.72 \tag{4-22}$$

$$c_1 + 0.4493 c_2 = 0.63 \tag{4-23}$$

$$c_1 + 0.3329 c_2 = 0.6 \tag{4-24}$$

$$c_1 + 0.2019 c_2 = 0.55 \tag{4-25}$$

$$c_1 + 0.1003 c_2 = 0.5 \tag{4-26}$$

在 MATLAB 命令窗口输入如下程序：

```
>>t = [0  0.3  0.81  1.1  1.6  2.3];        %输入时间向量 t
>>y = [0.82  0.72  0.63  0.6  0.55  0.5];   %输入向量 y
>>plot(t,y,'ro')                            %绘制测试数据的散点图,标记点为红色圆点
```

```
≫hold on                                  %图形保持
≫c0 = exp(- t')                           %计算衰减指数向量
c0 =
    1.0000
    0.7408
    0.4449
    0.3329
    0.2019
    0.1003
≫A = [ones(6,1), c0]                      %输入方程组的系数矩阵
A =
    1.0000    1.0000
    1.0000    0.7408
    1.0000    0.4493
    1.0000    0.3329
    1.0000    0.2019
    1.0000    0.1003
≫b = y'                                   %输入方程组等式右侧列向量
b =
    0.8200
    0.7200
    0.6300
    0.6000
    0.5500
    0.5000
≫C = pinv(A)* b                           %解超定方程组
C =
    0.4759
    0.3413
≫t = 0:0.05:2.4;                          %设置时间向量
≫y1 = C(1) + C(2)* exp(- t);              %计算拟合曲线输出值
≫plot(t,y1,'- k')                         %绘制拟合曲线线性图,线型为黑色实线
≫grid on                                  %加上坐标栅格线。如图 4-2 所示
```

【例 4-6】 某城市人均年收入与人均年消费经分类平均后，有 12 个样本如表 4-2 所示，试求出二次回归方程式，相关系数和回归误差。

表 4-2 人均年收入与人均年消费分类平均后的 12 个样本

样 本 号	1	2	3	4	5	6	7	8	9	10	11	12
人均年收入/万元	0.8	1.0	1.2	1.4	1.6	1.8	2.0	2.5	3.0	3.5	4.0	5.0
人均年消费/万元	0.65	0.7	0.77	0.85	0.98	1.10	1.25	1.45	1.73	1.95	2.32	3.0

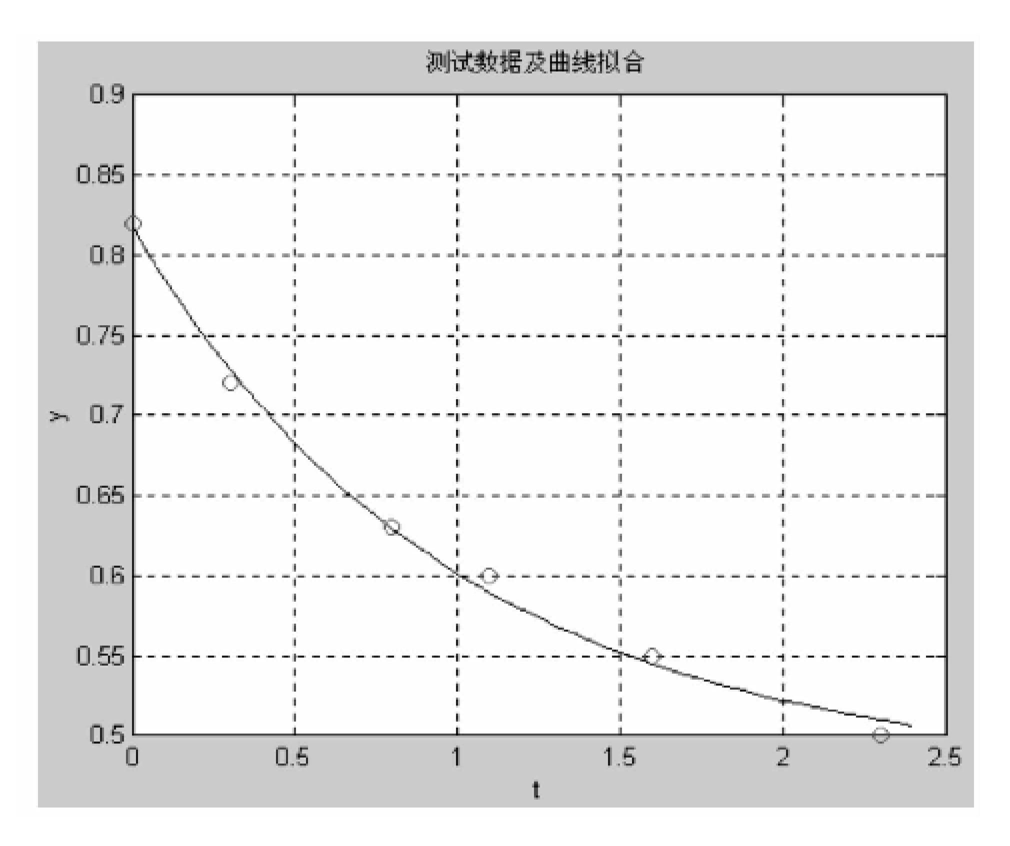


图 4-2 测试数据散点图及曲线拟合

设二次回归方程式为：

$$y = c_1 + c_2 x + c_3 x^2 \tag{4-27}$$

式中，y——年人均消费（万元计），x——年人均收入（万元计），c_1、c_2、c_3——二次回归系数。

解：

以向量 x 的元素，逐个代入式（4-27）得下列 12 个方程式

$$\left.\begin{aligned}
c_1 + 0.8c_2 + 0.64c_3 = 0.65 \\
c_1 + 1.0c_2 + c_3 = 0.7 \\
c_1 + 1.2c_2 + 1.44c_3 = 0.77 \\
c_1 + 1.4c_2 + 1.96c_3 = 0.85 \\
c_1 + 1.6c_2 + 2.56c_3 = 0.98 \\
c_1 + 1.8c_2 + 3.24c_3 = 1.10 \\
c_1 + 2.0c_2 + 4.00c_3 = 1.25 \\
c_1 + 2.5c_2 + 6.25c_3 = 1.45 \\
c_1 + 3.0c_2 + 9.00c_3 = 1.73 \\
c_1 + 3.5c_2 + 12.25c_3 = 1.95 \\
c_1 + 4.0c_2 + 16.00c_3 = 2.32 \\
c_1 + 5.0c_2 + 25.00c_3 = 30.00
\end{aligned}\right\} \tag{4-28}$$

写成矩阵形式为

$$\boldsymbol{AC} = \boldsymbol{b} \tag{4-29}$$

式中 $\boldsymbol{A}$ 为系数矩阵，$\boldsymbol{C} = [c_1; c_2; c_3]$为待求向量，$\boldsymbol{b}$ 为常数项向量。所以

$$C = pinv(A)*b \tag{4-30}$$

这是一组含有 3 个未知数，12 个方程式的超定方程组，为了求解 c_1、c_2、c_3，在 MATLAB 命令窗口输入如下程序：

```
≫x = [0.8 1.0 1.2 1.4 1.6 1.8 2 2.5 3 3.5 4 5]';   %输入年收入向量
≫y = [0.65 0.7 0.77 0.85 0.98 1.1 1.22 1.45 1.73 1.95 2.32 3.0]';
                                                  %输入年消费向量
≫plot(x,y,'ro')                                   %绘制收入,消费散点图,标记点设为红
                                                    色圆点,如图 4-3 所示
```

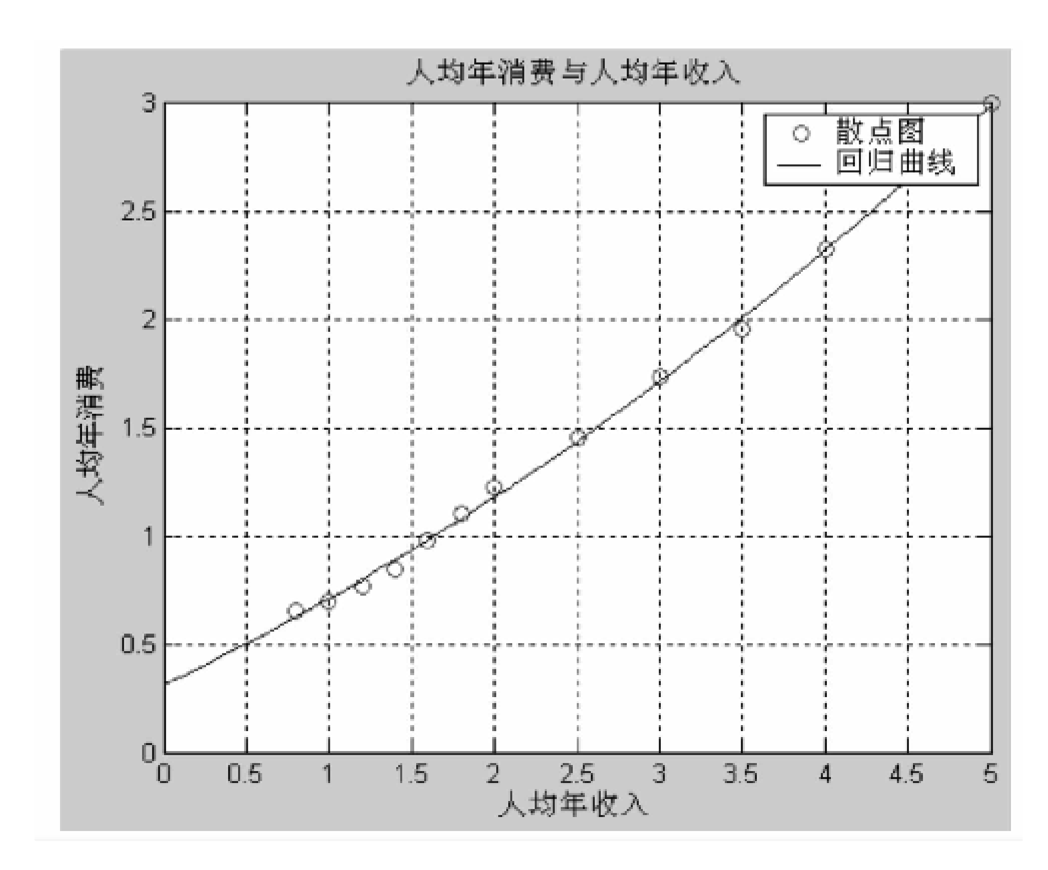


图 4-3 人均年消费与人均年收入

```
≫hold on                                          %设置图形保持
≫A = [ones(12,1),x,x.^2]                          %输入方程组(4-28)的系数矩阵
A =

    1.0000    0.8000    0.6400
    1.0000    1.0000    1.0000
    1.0000    1.2000    1.4400
    1.0000    1.4000    1.9600
    1.0000    1.6000    2.5600
    1.0000    1.8000    3.2400
    1.0000    2.0000    4.0000
```

```
     1.0000    2.5000    6.2500
     1.0000    3.0000    9.0000
     1.0000    3.5000   12.2500
     1.0000    4.0000   16.0000
     1.0000    5.0000   25.0000
>>b = y;                                    %输入方程组(4-28)的常数项向量
>>C = pinv(A) * b                           %解方程组(4-28)
C =
    0.3110                                  %无收入者年生活费用最低标准估计值，
                                             即 c1
    0.3648                                  %收入增加引起消费增加的 1 次项系数
    0.0342                                  %收入增加引起消费增加的 2 次项系数
>>x1 = 0:0.1:5;                             %为绘制回归曲线而设置收入向量
>>y1 = C(1) + C(2) * x1 + C(3) * x1.^2;     %为绘制回归曲线的消费向量
>>plot(x1,y1,'-k')                          %绘制回归曲线
>>corrcoef(x,y)                             %相关系数为 0.9966
ans =
    1.0000    0.9966
    0.9966    1.0000
>>E = b - C(1) - C(2) * x - C(3) * x.^2;    %误差向量
E'
ans =
  Columns 1 through 10
    0.0254   -0.0099   -0.0279   -0.0386   -0.0021    0.0217    0.0428
0.0135    0.0172   -0.0563
  Columns 11 through 12
    0.0032    0.0109
```

4.3 欠定方程组

当线性方程式的数量（m）小于变量数（n）时，则称为欠定方程组。欠定方程组的求解可以分为两步。首先计算方程组的特解。特解可以从矩阵左除法得到，即

$$X = A \backslash b \tag{4-31}$$

式中，A 为系数矩阵，b 为常数项向量。特解中有 m 个非零元素和（n - m）个零元素。
第二步是求补解 Xd，补解

$$Xd = null(A,'r') * q \tag{4-32}$$

式中函数 null (A,'r')返回 1 个矩阵 A 的零空间的正交基。所谓正交基是指，若 z = null (A,'r') 则

$$z' * z = eye(rank(A)) \tag{4-33}$$

所谓零空间是指 A* z 可忽略不计。式(4-32)中的“r”是指实数。
式(4-32)中的 q 为任意列向量，但它的列数必需等于 z 的行数。
下面举例于以说明。

【例 4-7】 已知欠定方程组如下所列：

$$17x_1+24x_2+x_3+8x_4+15x_5=110 \tag{4-34}$$

$$23x_1+5x_2+7x_3+14x_4+16x_5=81 \tag{4-35}$$

$$4x_1+6x_2+13x_3+20x_4+22x_5=82 \tag{4-36}$$

求此欠定方程组的特解和通解。

解：

在 MATLAB 命令窗口输入如下程序：

```
≫A = [17 24 1 8 15;23 5 7 14 16;4 6 13 20 22]        %输入系数矩阵
A =

    17    24     1     8    15
    23     5     7    14    16
     4     6    13    20    22

≫b = [110; 81; 82]                                    %输入常数项向量
b =
   110
    81
    82
≫X = A \ b                                            %用左除法求方程组的特解，其中有两
                                                        个零元素

X =

   1.0000
   2.0000
        0
        0
   3.0000
≫z = null (A,'r')                                     %求矩阵A的零空间正交基，所以有无
                                                        穷个解

z =

   -0.1401   -0.1610
   -0.1751   -0.4512
   -1.4145   -1.4345
    1.0000         0
         0    1.0000
```

欠定方程式的通解为

```
X0 = X + z* q                                          %q为2列的任意向量
```

```
≫q = [1; 2];                          %现设为 [1; 2]
≫Xd = z* q                            %在特定 q 时的补解
Xd =

   -0.4621
   -1.0776
   -4.2836
    1.0000
    2.0000

≫X0 = X + Xd                          %在特定 q 时的解
X0 =

    0.5379
    0.9224
   -4.2836
    1.0000
    5.0000

≫A1* X0                               %以 X0 代入，核对结果是正确的，尾数
                                        为计算误差
ans =

  110.0001
   81.0001
   82.0002
```

【例 4-8】 传说中在一次沉船事故中，有 5 个人和 1 只猴子脱险，他们被漂流到荒岛上，为了生存，他们采集了许多椰子。夜里，其中 1 人先爬起来，企图多分椰子，为此他把椰子分成 5 份，自己先藏走 1 份，余下 1 个给猴子，随后继续睡觉。接着，第 2、3、4、5 位遇难者相继爬起来，以同样的动机做了同样的事。清晨，他们全起来后，共同把椰子分成 5 份，每人各得 1 份，恰好仍余下 1 只给了猴。试问他们共采集了多少只椰子，每人各分得多少只椰子。

解：

设 5 人私分的椰子数分别为 x_1、x_2、x_3、x_4、x_5，而共分的椰子数为 y，则根据题意应满足下列方程组：

$$4x_1 - 5x_2 = 1 \tag{4-37}$$

$$4x_2 - 5x_3 = 1 \tag{4-38}$$

$$4x_3 - 5x_4 = 1 \tag{4-39}$$

$$4x_4 - 5x_5 = 1 \tag{4-40}$$

$$4x_5 - 5y = 1 \tag{4-41}$$

这是 1 组欠定线性方程组，它有无穷多个解。但若求最小正整数解，则有惟一解。从式（4-41）可知

$$x_5 = y + (y+1)/4 \tag{4-42}$$

若 x_5 为正整数，则

$$y = 4n - 1 \tag{4-43}$$

式中，n 为正整数，山此得

$$4x_5 = 20n - 4 \tag{4-44}$$

将 n 视为可变正整数，搜索待求变量的整数解即可，联立解式（4-37）、式（4-38）、式（4-39）、式（4-40）、式（4-44），即可得线性方程组的解。在命令窗口输入下列语句（下列语句亦可用 M 文件编写，调试并运行。）：

```
≫A = [4  -5 0 0 0;0 4  -5 0 0;0 0 4  -5 0;0 0 0 4  -5;0 0 0 0 4];
                                                    %方程组的系数矩阵
≫for n = 1:1000;                                    %设置循环语句,参数为 n
      b = [1 1 1 1 20* n - 4]';                     %设置方程组右侧列向量
      x = inv(A)* b;                                %方程组的解
      if x(5) = = fix(x(5)) & x(4) = = fix(x(4)) & x(3) = = fix(x(3)) &…
                                                    %整数解条件
         x (2) = = fix (x (2)) & x (1) = = fix (x (1))
      break                                         %符合整数解条件，则中断循环
      end;                                          %结束条件判断
   end;                                             %否则，将 n 加 1 再进行循环
≫ x                                                 %显示方程解
x =
      3124                                          %即为 x1
      2499                                          %即为 x2
      1999                                          %即为 x3
      1599                                          %即为 x4
      1279                                          %即为 x5
≫ n                                                 %显示循环参数实际值
n =
   256
```

将 n 代入式（4-43）则得

y = 1023

由此得椰子总数为

N = x1 + x2 + x3 + x4 + x5 + 5* y + 6

= 15621

每人分得椰子数为：

```
x1 + y = 4147                                       %第 1 人
x2 + y = 3522                                       %第 2 人
x3 + y = 3022                                       %第 3 人
x4 + y = 2622                                       %第 4 人
```

x5 + y = 2302　　　　　　　　　　　　　　　　%第 5 人
猴子分得椰子数为 6 个　　　　　　　　　　　　%每人私分时得 5 个，共分时又得 1 个

4.4 代数方程式的符号解

上面介绍的是线性方程组，理论上对任意个变量的线性方程组在符合一定的条件下，都能找到一般解。但对于 2 次或 2 次以上的代数方程式或非线性方程组，就很难得到一般解。MATLAB 在引进 Maple 数学软件中的符号变量后，采用代数方程符号解函数 solve，就使得解代数方程式变得容易。代数方程符号解函数 solve 的书写格式如下：

g = solve(eq)　　　　(4-45)

g = solve(eq, var)　　　　(4-46)

g = solve(eq1, eq2, …, eqn)　　　　(4-47)

g = solve(eq1, eq2, …, eqn, var1, var2, …, varn)　　　　(4-48)

式中，eqi(i = 1, 2, …, n)为数学符号表达式或字符串表达式，并且约定等于零；vari(I = 1, 2, …, n)为需要求解的变量。现举例说明它的应用。

【例 4-9】　求解二元一次线性方程组

$$\begin{cases} a_{11}x + a_{12}y = b_1 & (4\text{-}49) \\ a_{21}x + a_{22}y = b_2 & (4\text{-}50) \end{cases}$$

解：

在命令窗口输入如下程序：

```
>>syms a11 a12 a21 a22 b1 b2 x y          %设置符号变量
>>eq1 = a11* x + a12* y - b1;             %设置符号表达式
>>eq2 = a21* x + a22* y - b2;             %设置符号表达式
>>S = solve(eq1, eq2)                     %解代数方程式
S =                                       %解的结果为结构数组
    x: [1x1 sym]
    y: [1x1 sym]
>>S.x                                     %符号变量 x 的解
ans =
-(-a22* b1 + b2* a12)/(a11* a22 - a12* a21)
>>S.y                                     %符号变量 y 的解
ans =
(a11* b2 - b1* a21)/(a11* a22 - a12* a21)
```

【例 4-10】　已知星形电路（见图 4-4）的电阻为 r_1、r_2 和 r_3，试将其转换成三角形电路，相应的电阻为 x_1、x_2 和 x_3，求 x_1、x_2 和 x_3 的符号解。若 $r_1 = 10\Omega$、$r_2 = 20\Omega$、$r_3 = 30\Omega$，求 x_1、x_2 和 x_3 的电阻值。

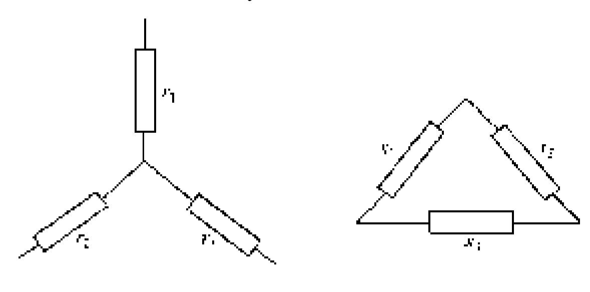


图 4-4　星/三角电路转换

解：

```
≫syms r1 r2 r3 x1 x2 x3 eq1 eq2 eq3;                    %设置符号变量
≫eq1 = r1 + r2 - x1* (x2 + x3)/(x1 + x2 + x3);          %电阻等值方程式
≫eq2 = r1 + r3 - x2* (x1 + x3)/(x1 + x2 + x3);          %电阻等值方程式
≫eq3 = r2 + r3 - x3* (x1 + x2)/(x1 + x2 + x3);          %电阻等值方程式
≫[x1,x2,x3] = solve(eq1,eq2,eq3)                        %解 eq1、eq2、eq3 方程式
x1 =                                                    %三角形等值电路中的 x1
(r1* r2 + r3* r2 + r1* r3)/r3
x2 =                                                    %三角形等值电路中的 x2
(r1* r2 + r3* r2 + r1* r3)/r2
x3 =                                                    %三角形等值电路中的 x3
(r1* r2 + r3* r2 + r1* r3)/r1
≫r1 = 10;r2 = 20;r3 = 30;                               %设星形电路的电阻值
≫x1 = eval(x1)                                          %执行字符串计算,得三角形电路的等值
                                                         电阻,eval 为转换函数
x1 =
    36.6667
≫x2 = eval(x2)
x2 =
    55
≫x3 = eval(x3)
x3 =
    110
```

用同样的方法可以进行三角形到星形的反转换。

【例 4-11】 已知 3 定点 d (2, 3), e (3, −1) 和 f (−3, −4) 求通过此 3 定点的圆方程式。

解:

设圆方程式的圆中心为 (a, b), 半径为 r, 则此圆应满足下列方程组:

$$\begin{cases}(2-a)^2+(3-b)^2-r^2=0 & (4\text{-}51)\\ (3-a)^2+(-1-b)^2-r^2=0 & (4\text{-}52)\\ (-3-a)^2+(-4-b)^2-r^2=0 & (4\text{-}53)\end{cases}$$

根据方程式 (4-51)、式 (4-52)、式 (4-53) 在 MATLAB 命令窗口写入下列程序:

```
≫syms a b r eq1 eq2 eq3;                                %设字符串变量
≫eq1 = (2 - a)^2 + (3 - b)^2 - r^2;                      %通过 d 点的圆方程式
≫eq2 = (3 - a)^2 + (- 1 - b)^2 - r^2;                    %通过 e 点的圆方程式
≫eq3 = (- 3 - a)^2 + (- 4 - b)^2 - r^2;                  %通过 f 点的圆方程式
≫[a,b,r] = solve(eq1,eq2,eq3)                           %解方程式(4-51)~式(4-53)
a =                                                     %圆心横坐标
[- 23/18]
[- 23/18]
```

```
b =                                                %圆心纵坐标
[1/18]
[1/18]
r =                                                %圆半径
[1/18*6290^(1/2)]
[-1/18*6290^(1/2)]
>>a = eval(a)                                      %用数值表示
a =
   -1.2778
   -1.2778
>>b = eval(b)                                      %用数值表示
b =
   0.0556
   0.0556
>>r = eval(r)                                      %用数值表示
r =                                                %用数值表示,取绝对值
   4.4061
```

故圆方程式为

$$(x+1.2778)^2+(y-0.0556)^2=(4.4061)^2$$

上述命令也可以用 M 文件编写通用的，求解通过已知 3 定点的圆方程程序。该程序解出圆方程的半径和圆心坐标。其源程序如下，函数名为 tripointc

```
function z = tripointc(x1,x2,x3)                   %函数定义项
%TRIPOINTC Three point circle equation's coordinate.  %帮助文本
%This program for solve the circle equation which passed three point.
%Input three point's coordinate into the parenthesis of function.
%Example: The three point is [-5,0],[5,0] and [0,10],Input as follows,then execute.
%>>tripointc([-5,0],[5,0],[0,10])
%          a =
%            0
%          b =
%            3.7500
%          r =
%            6.2500
% See also solve,eval,syms,abs
syms a b r                                         %程序体
eq1 = (x1(1) - a).^2 + (x1(2) - b).^2 - r.^2;
eq2 = (x2(1) - a).^2 + (x2(2) - b).^2 - r.^2;
eq3 = (x3(1) - a).^2 + (x3(2) - b).^2 - r.^2;
[a,b,r] = solve(eq1,eq2,eq3);
```

```
a = eval(a(1))
b = eval(b(1))
r = abs(eval(r(1)))
```

【例 4-12】 解下列 3 次代数方程式的根

$$x^3 + 5x^2 + 24x + 20 = 0 \tag{4-54}$$

解：

```
≫solve('x^3 + 5* x^2 + 24* x + 20')          %输入代数方程式表达式，并求解
ans =                                          %答，有 1 个实根和一对复根
[        -1]
[-2 + 4* i]
[-2 - 4* i]
```

【例 4-13】 解代数方程组：

$$\begin{cases} (x/3)^2 + (y/4)^2 - 1 = 0 & (4\text{-}55) \\ y = 2x^2 & (4\text{-}56) \end{cases}$$

解：

在命令窗口输入如下程序：

```
≫syms x y                                  %设字符串变量
≫eq1 = (x/3)^2 + (y/4)^2 - 1;              %将(4-55)式赋值给变量 eq1
≫eq2 = y - 2* x^2;                         %将(4-56)式赋值给变量 eq2
≫[x,y] = solve(eq1,eq2)                    %解联立方程式 eq1,eq2
x =                                         %x 的解
[ 1/3* (-2 + 2* 82^(1/2))^(1/2)]
[-1/3* (-2 + 2* 82^(1/2))^(1/2)]
[ 1/3* (-2 - 2* 82^(1/2))^(1/2)]
[-1/3* (-2 - 2* 82^(1/2))^(1/2)]
y =                                         %y 的解
[ -4/9 + 4/9* 82^(1/2)]
[ -4/9 + 4/9* 82^(1/2)]
[ -4/9 - 4/9* 82^(1/2)]
[ -4/9 - 4/9* 82^(1/2)]
≫x = eval(x)                               %转换字符串为数值解
x =

    1.3379
   -1.3379
    0.0000   + 1.4948i
   -0.0000   - 1.4948i

≫y = eval(y)                               %转换字符串为数值解
y =
```

```
  3.5802
  3.5802
 -4.4691
 -4.4691
```

【例 4-14】 已知 3 变量代数方程组如下：

$$\begin{cases} x^2 + y^2 = 1 & (4\text{-}57) \\ y + xz = 0 & (4\text{-}58) \\ yz - \sqrt{2}/2 = 0 & (4\text{-}59) \end{cases}$$

求 x、y、z 的解。

解：

在 MATLAB 命令窗口中输入如下程序：

```
>>syms x y z                                    %定义符号变量
>>eq1 = sqrt(x.^2 + y.^2) - 1;                  %写入方程式(4-57),sqrt为根号
>>eq2 = y + x* z;                               %写入方程式(4-58)
>>eq3 = x + y* z - sqrt(2)/2;                   %写入方程式(4-59)
>>S = solve(eq1,eq2,eq3)                        %解方程组
S =                                             %显示解的结构
    x:[4x1 sym]
    y:[4x1 sym]
    z:[4x1 sym]
>> S.x                                          %符号变量x的解
ans =
[-1/4* 2^(1/2)* (-1/2 + 1/2* 17^(1/2))]
[-1/4* 2^(1/2)* (-1/2 + 1/2* 17^(1/2))]
[-1/4* 2^(1/2)* (-1/2 - 1/2* 17^(1/2))]
[-1/4* 2^(1/2)* (-1/2 - 1/2* 17^(1/2))]
>>x = eval(S.x)                                 %转换符号变量x为数值
x =
  -0.5521
  -0.5521
   0.9056
   0.9056
>>S.y                                           %符号变量y的解
ans =
[  1/8* 2^(1/2)* (-1/2 + 1/2* 17^(1/2))* (5 + 17^(1/2))^(1/2)]
[-1/8* 2^(1/2)* (-1/2 + 1/2* 17^(1/2))* (5 + 17^(1/2))^(1/2)]
[  1/8* 2^(1/2)* (-1/2 - 1/2* 17^(1/2))* (5 - 17^(1/2))^(1/2)]
[-1/8* 2^(1/2)* (-1/2 - 1/2* 17^(1/2))* (5 - 17^(1/2))^(1/2)]
```

```
≫ y = eval(S.y)                                  %转换符号变量 y 为数值
y  =
         0.8338
       - 0.8338
       - 0.4240
         0.4240
≫S.z                                             %符号变量 z 的解
ans =
[   1/2* (5 + 17^(1/2))^(1/2)]
[- 1/2* (5 + 17^(1/2))^(1/2)]
[   1/2* (5 - 17^(1/2))^(1/2)]
[- 1/2* (5 - 17^(1/2))^(1/2)]
≫z = eval(ans)                                   %转换符号变量 z 为数值
z =
         1.5102
       - 1.5102
         0.4682
       - 0.4682
≫answer = [x, y, z]                              % 代数方程组的 4 组解,每行代表 1 组
                                                   解, x 代表第一列, y 代表第二列, z
                                                   代表第三列
answer =
       - 0.5521      0.8338      1.5102
       - 0.5521    - 0.8338    - 1.5102
         0.9056    - 0.4240      0.4682
         0.9056      0.4240      0.4682
```

【例 4-15】 已知某系统的方框图如图 4-5 所示。

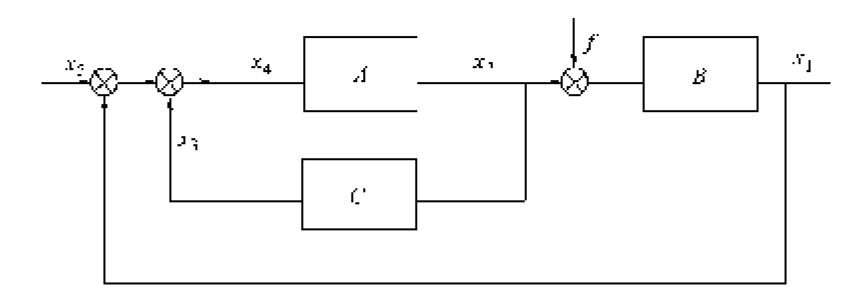


图 4-5 某系统方框图

其变量之间的关系如下:

$$x_1 = B(x_2 - f) \tag{4-60}$$

$$x_2 = A x_4 \tag{4-61}$$

$$x_3 = Cx_2 \tag{4-62}$$

$$x_4 = x_5 - x_1 - x_3 \tag{4-63}$$

式中，A、B、C 为系统中的传递函数，求闭环传递函数，即 x_1 与 x_5 之比。

解：

在 MATLAB 命令窗口输入下列程序：

```
≫syms x1 x2 x3 x4 x5 f                   %设置控制变量，x1、x2、x3、x4、x5、f 为符号变量
≫syms A B C                              %设置传递函数 A，B，C 为符号变量
≫eq1 = x1 - B*(x2 - f);                  %将（4-60）式赋值给 eq1
≫eq2 = x2 - A*x4;                        %将（4-61）式赋值给 eq2
≫eq3 = x3 - C*x2;                        %将（4-62）式赋值给 eq3
≫eq4 = x4 + x3 - (x5 - x1);              %将（4-63）式赋值给 eq4
≫S = solve(eq1,eq2,eq3,eq4)              %解方程组
S =                                      %解的结构形式
    x1: [1x1 sym]
    x2: [1x1 sym]
    x3: [1x1 sym]
    x4: [1x1 sym]
≫S.x1                                    %控制变量 x1 的解，即为所求
ans =
-B*(f + C*A*f - x5*A)/(1 + C*A + B*A)
```

【例 4-16】 x1，x2∈[0，π/2]，解下列三角函数方程组

$$eq_1 = \cos^2 x_1 - \sin^2 x_1 - \cos x_1 \cos x_2 = 0 \tag{4-64}$$

$$eq_2 = \cos^2 x_2 - \sin^2 x_2 - \sin x_1 \sin x_2 = 0 \tag{4-65}$$

解：

在 MATLAB 窗口输入如下程序：

```
≫syms x1 x2                                      %设符号变量 x1、x2
≫eq1 = cos(x1)^2 - sin(x1)^2 - cos(x1)*cos(x2);  %输入式（4-64），赋予 eq1
≫eq2 = cos(x2)^2 - (sin(x2) - sin(x1))*sin(x2);  %输入式（4-65），赋予 eq2
≫ [x1, x2] = solve (eq1, eq2)                    %解三角方程式
x1 =
[ atan(1/10*(50 + 10*5^(1/2))^(1/2)/(3/10*(50 + 10*5^(1/2))^(1/2) - 1/200*(50 + 10*5^(1/2))^(3/2))) + pi]
[ -atan(1/10*(50 + 10*5^(1/2))^(1/2)/(-3/10*(50 + 10*5^(1/2))^(1/2) + 1/200*(50 + 10*5^(1/2))^(3/2)))]
[ atan(1/10*(50 - 10*5^(1/2))^(1/2)/(3/10*(50 - 10*5^(1/2))^(1/2) - 1/200*(50 - 10*5^(1/2))^(3/2)))]
[-atan(1/10*(50 - 10*5^(1/2))^(1/2)/(-3/10*(50 - 10*5^(1/2))^(1/2) + 1/200*(50 -
```

```
10* 5^(1/2))^(3/2))) - pi]
    [      atan(1/10* (50 + 10* 5^(1/2))^(1/2)/(- 3/10* (50 + 10* 5^(1/2))^(1/2) + 1/200*
(50 + 10* 5^(1/2))^(3/2)))]
    [  - atan(1/10* (50 + 10* 5^(1/2))^(1/2)/(3/10* (50 + 10* 5^(1/2))^(1/2) - 1/200* (50 +
10* 5^(1/2))^(3/2))) - pi]
    [  atan(1/10* (50 - 10* 5^(1/2))^(1/2)/(- 3/10* (50 - 10 * 5^(1/2))^(1/2) + 1/200* (50 -
10* 5^(1/2))^(3/2))) + pi]
    [     - atan(1/10* (50 - 10* 5^(1/2))^(1/2)/(3/10* (50 - 10* 5^(1/2))^(1/2) - 1/200* (50
- 10* 5^(1/2))^(3/2)))]
    [                                    atan((54 + 6* 33^(1/2))^(1/2)/(90 - 6*
33^(1/2))^(1/2))]
    [                                  - atan((54 + 6* 33^(1/2))^(1/2)/(90 - 6* 33^(
1/2))^(1/2)) + pi]
    [                                   - atan((54 + 6* 33^(1/2))^(1/2)/(90 - 6*
33^(1/2))^(1/2))]
    [                                  atan((54 + 6* 33^(1/2))^(1/2)/(90 - 6* 33^
(1/2))^(1/2)) - pi]
    [                                    atan((54 - 6* 33^(1/2))^(1/2)/(90 + 6*
33^(1/2))^(1/2))]
    [                                  - atan((54 - 6* 33^(1/2))^(1/2)/(90 + 6* 33^(
1/2))^(1/2)) + pi]
    [                                   - atan((54 - 6* 33^(1/2))^(1/2)/(90 + 6*
33^(1/2))^(1/2))]
    [                                  atan((54 - 6* 33^(1/2))^(1/2)/(90 + 6* 33^
(1/2))^(1/2)) - pi]
    x2 =
    [
                    - atan(- 1/2 + 1/2* 5^(1/2))]
    [
                  - atan(- 1/2 + 1/2* 5^(1/2)) + pi]
    [
                      atan(1/2* 5^(1/2) + 1/2)]
    [
                    atan(1/2* 5^(1/2) + 1/2) - pi]
    [
                    atan(- 1/2 + 1/2* 5^(1/2)) - pi]
    [
                      atan(- 1/2 + 1/2* 5^(1/2))]
    [
```

```
                    -atan(1/2*5^(1/2)+1/2)+pi]
[
                     -atan(1/2*5^(1/2)+1/2)]
[ atan(1/24*(54+6*33^(1/2))^(1/2)*(-1/2+1/2*33^(1/2))/(-1/576*(90-6*33^(1/2))^(1/2)*(54+6*33^(1/2))+5/48*(90-6*33^(1/2))^(1/2)))+pi]
[     atan(1/24*(54+6*33^(1/2))^(1/2)*(-1/2+1/2*33^(1/2))/(1/576*(90-6*33^(1/2))^(1/2)*(54+6*33^(1/2))-5/48*(90-6*33^(1/2))^(1/2)))]
[-atan(1/24*(54+6*33^(1/2))^(1/2)*(-1/2+1/2*33^(1/2))/(-1/576*(90-6*33^(1/2))^(1/2)*(54+6*33^(1/2))+5/48*(90-6*33^(1/2))^(1/2)))-pi]
[     -atan(1/24*(54+6*33^(1/2))^(1/2)*(-1/2+1/2*33^(1/2))/(1/576*(90-6*33^(1/2))^(1/2)*(54+6*33^(1/2))-5/48*(90-6*33^(1/2))^(1/2)))]
[     atan(1/24*(54-6*33^(1/2))^(1/2)*(-1/2-1/2*33^(1/2))/(-1/576*(90+6*33^(1/2))^(1/2)*(54-6*33^(1/2))+5/48*(90+6*33^(1/2))^(1/2)))]
[   atan(1/24*(54-6*33^(1/2))^(1/2)*(-1/2-1/2*33^(1/2))/(1/576*(90+6*33^(1/2))^(1/2)*(54-6*33^(1/2))-5/48*(90+6*33^(1/2))^(1/2)))-pi]
[   -atan(1/24*(54-6*33^(1/2))^(1/2)*(-1/2-1/2*33^(1/2))/(-1/576*(90+6*33^(1/2))^(1/2)*(54-6*33^(1/2))+5/48*(90+6*33^(1/2))^(1/2)))]
[ -atan(1/24*(54-6*33^(1/2))^(1/2)*(-1/2-1/2*33^(1/2))/(1/576*(90+6*33^(1/2))^(1/2)*(54-6*33^(1/2))-5/48*(90+6*33^(1/2))^(1/2)))+pi]
>>x1=eval(x1);                          %将字符串转换成数值
>>x2=eval(x2);                          %将字符串转换成数值
>>x1'                                   %列向量 x1 以行形式显示
ans =
  Columns 1 through 11
    2.1244   -1.0172    0.5536   -2.5880    1.0172   -2.1244    2.5880   -0.5536
0.9008    2.2408   -0.9008
  Columns 12 through 16
   -2.2408    0.3772    2.7644   -0.3772   -2.7644
>>x2'                                   %列向量 x1 以行形式显示
ans =
  Columns 1 through 11
   -0.5536    2.5880    1.0172   -2.1244   -2.5880    0.5536    2.1244   -1.0172
1.9480    1.1936   -1.9480
  Columns 12 through 16
   -1.1936   -0.6700   -2.4716    0.6700    2.4716
>>x1=0.5536                             %符合小于 pi/2 的解 x1
x1 =
    0.5536
>>x2=1.0172                             %符合小于 pi/2 的解 x1
```

```
x2 =
    1.0172
≫degx1 = x1* 180/pi                          %x1 以度数表示
degx1 =
    31.7189
≫degx2 = x2* 180/pi                          %x2 以度数表示
degx2 =
    58.2813
```

4.5 线性方程组的迭代解法之一：Jacobian 迭代法

线性方程组的解法，基本上可分为直接解法和迭代解法。直接解法中有高斯消去法，L，U 分解法和平方根分解法等。但随着阶次的增加，计算机所占内存越来越大，且运算时间也大为增加。对于系数矩阵为稀疏矩阵的线性方程组，采用迭代解法，有时是有益的。

迭代解法是构造一个向量序列 $\boldsymbol{x}_k$，通过有限次的运算，使它收敛于某个极限 x。它占用计算机的内存较少。计算方法也较简单。但关键问题是收敛问题，以及收敛速度问题。下面介绍一个古典迭代解法，即 Jacobian 迭代法。

考虑非奇异线性方程式组

$$\boldsymbol{AX}=\boldsymbol{b} \tag{4-66}$$

将 $\boldsymbol{A}$ 分解为

$$\boldsymbol{A}=\mathrm{D}-\mathrm{L}-\mathrm{U} \tag{4-67}$$

其中

$$\mathrm{D}=\mathrm{diag}\,(\mathrm{diag}\,(\mathrm{A})) \tag{4-68}$$

为对角矩阵。

$$\mathrm{L}=-\mathrm{tril}\,(\mathrm{A}) \tag{4-69}$$

为下三角矩阵。

$$\mathrm{U}=-\mathrm{triu}\,(\mathrm{A}) \tag{4-70}$$

为上三角矩阵。那么式（4-66）可写成

$$\boldsymbol{X}=\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})\boldsymbol{X}+\boldsymbol{D}^{-1}\boldsymbol{b} \tag{4-71}$$

写成迭代形式为

$$\boldsymbol{X}_k=\boldsymbol{D}^{-1}(\boldsymbol{L}+\boldsymbol{U})\boldsymbol{X}_{k-1}+\boldsymbol{D}^{-1}\boldsymbol{b} \tag{4-72}$$

式中，下标 $k=1，2，3，\cdots，n$。

这就是 Jacobian 迭代公式。把上述迭代公式编制成函数 M 文件如下：

```
%线性方程组的 Jacobian 迭代解法
% A——系数矩阵
% b——常数向量
% X0——初值
% n——迭代次数
% X——解向量
function X = fjacobi(A,b,X0)                  %函数定义项
D = diag(diag(A));                            %对角矩阵
L = -tril(A,-1);                              %下三角矩阵
```

```
U = -triu(A,1);                    %上三角矩阵
B = D \ (L+U);                     %迭代形式的系数矩阵
F = D \ b;                         %迭代形式的常数向量
X = B* X0 + F;                     %Jacobian 迭代公式
n = 1;
while norm(X-X0) >= 1e-5           % 当差向量的 2 范数大于万分之一时,继
                                     续迭代,否则结束迭代
    X0 = X;                        %上次解向量,赋值给下一个
    X = B* X0 + F; n = n+1;        %迭代运算,迭代次数加 1
    if n>100                       %若迭代次数大于 100 则中断
        break
    end
end
n                                  %显示迭代次数 n
```

关于迭代法能否收敛问题，它与初始向量 $\boldsymbol{X}_0$ 无关。主要取决于迭代矩阵$\boldsymbol{B}=\boldsymbol{D}\setminus(\boldsymbol{L}+\boldsymbol{U})$的特征值 λ_i。若 $\max|\lambda_i|<1$，则迭代收敛，否则将是发散的。有关证明，请查阅数学手册。

【例 4-17】 用迭代法解下列线性方程组。

$$\begin{cases}7x_1-3x_2+2x_3=17\\4x_1+9x_2-x_3=29\\6x_1+3x_2+11x_3=35\end{cases}$$

解：

```
>>A = [7 -3 2;4 9 -1;6 3 11]       %输入系数矩阵 A
A =

     7    -3     2
     4     9    -1
     6     3    11

>>b = [17;29;35]                   %输入常数项向量
b =
    17
    29
    35
>>X0 = [1;1;1];                    %设置初值
>>fjacobi(A,b,X0)                  %调用 Jacobianr 迭代公式
n =                                %迭代次数
    16
ans =                              %解答
    3.0000
    2.0000
```

```
    1.0000
>>X = A \ b                                   %用矩阵左除法,核对解题结果
X =
    3.0000
    2.0000
    1.0000
```

【例 4-18】 已知 7 阶方阵 $\boldsymbol{A}$，主对角线元素均为 8，主对角线下一行元素均为 -5，主对角线上一行元素均为 -3，常数行向量 $\boldsymbol{b}$ = [2 3 4 5 6 7 8]，求解线性方程组 $\boldsymbol{AX}=\boldsymbol{b}$。

解：

```
>>V0 = 8* ones(1,7)                           %设置对角线向量
V0 =
    8    8    8    8    8    8    8
>>V1 = -3* ones(1,6)                          %设置上一行对角线向量
V1 =
    -3    -3    -3    -3    -3    -3
>>Vn1 = -5* ones(1,6)                         %设置下一行对角线向量
Vn1 =
    -5    -5    -5    -5    -5    -5
>>A = diag(V0) + diag(V1,1) + diag(Vn1,-1)    %创建方程组的系数矩阵 A
A =
     8    -3     0     0     0     0     0
    -5     8    -3     0     0     0     0
     0    -5     8    -3     0     0     0
     0     0    -5     8    -3     0     0
     0     0     0    -5     8    -3     0
     0     0     0     0    -5     8    -3
     0     0     0     0     0    -5     8
>>b = [2,3,4,5,6,7,8]';                       %输入常数项向量
>>X0 = [0;0;0;0;0;0;0];                       %设置解的初始值
>>fjacobi(A,b,X0)                             %调用 Jacobian 迭代法
n =                                           %迭代次数
    151
ans =                                         %线性方程组的迭代解
    1.4311
    3.1496
    5.0139
    6.7875
    8.0770
    8.2261
```

```
    6.1413
≫X = A \ b                                      %用矩阵左除法来核对解答的正确性
X =
    1.4311
    3.1496
    5.0139
    6.7875
    8.0770
    8.2261
    6.1413
```

【例 4-19】 已知两组线性方程组 $\boldsymbol{AX}=\boldsymbol{b}$，$\boldsymbol{A}_1\boldsymbol{X}_1=\boldsymbol{b}$，$\boldsymbol{A}=\begin{bmatrix}3&2&1\\2&4&0\\1&0&5\end{bmatrix}$，$\boldsymbol{A}_1=\begin{bmatrix}3&2&1\\3&4&3\\1&2&5\end{bmatrix}$，$\boldsymbol{b}=[2\ 3\ 4]$，若用 Jacobian 迭代法解方程组判断，哪个方程组收敛，哪个不收敛。

解：

```
≫A = [3 2 1;2 4 0;1 0 5]                       %输入矩阵 A
A =

     3     2     1
     2     4     0
     1     0     5

≫A1 = [3 2 1;3 4 3;1 2 5]                      %输入矩阵 A1
A1 =

     3     2     1
     3     4     3
     1     2     5

≫D = diag(diag(A))                             %提取对角矩阵
D =

     3     0     0
     0     4     0
     0     0     5

≫den = (-tril(A, -1) - triu(A,1))              %提取系数矩阵 A 中的上、下三角形
den =

     0    -2    -1
    -2     0     0
    -1     0     0

≫den1 = (-tril(A1, -1) - triu(A1,1))           %提取系数矩阵 A1 中的上、下三角形
den1 =
```

```
     0    -2    -1
    -3     0    -3
    -1    -2     0
>>B = D \ den                       %系数矩阵 A 的迭代式的系数矩阵 B
B =
          0   -0.6667   -0.3333
    -0.5000         0         0
    -0.2000         0         0
>>eig(B)                            %矩阵 B 的特征值,小于 1,故迭代式收敛
ans =
     0.6325
    -0.6325
          0
>>b = [2 3 4]';                     %输入常数项向量 b
>>X0 = [0;0;0];                     %输入初值
>>fjacobi(A,b,X0)                   %调用 Jacobian 迭代法
n =                                 %迭代次数
    59
ans =                               %迭代解
    -0.2308
     0.9231
     0.8462
>>X = A \ b                         %核对线性方程组的解
X =
    -0.2308
     0.9231
     0.8462
>>B1 = D \ den1                     %系数矩阵 A1 的迭代式的系数矩阵 B1
B1 =
          0   -0.6667   -0.3333
    -0.7500         0   -0.7500
    -0.2000   -0.4000         0
>>eig(B1)                           % 矩阵 B1 的特征值,大于 1,故迭代式不
                                      收敛
ans =
    -1.0300
```

```
        0.2485
        0.7815
>>A = A1;                                  %将 A1 赋予 A
>>fjacobi(A,b,X0)                          %调用 Jacobian 迭代法
n =                                        %超出最大迭代次数,不收敛
      501
ans =
   1.0e+006*
        1.0192
        1.2357
        0.6778
>>X = A \ b                                %用左除法的解
X =
        0.6000
       -0.3000
        0.8000
```

4.6 线性方程组的迭代解法之二：G-S迭代法

在采用 Jacobian 迭代法不收敛的情况下，还可以试用 Gauss – Seidel 迭代法。由式（4-66）得

$$(\boldsymbol{D}-\boldsymbol{L}-\boldsymbol{U})\boldsymbol{X}=\boldsymbol{b} \tag{4-73}$$

$$(\boldsymbol{D}-\boldsymbol{L})\boldsymbol{X}=\boldsymbol{U}\boldsymbol{X}+\boldsymbol{b} \tag{4-74}$$

$$\boldsymbol{X}=(\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{U}\boldsymbol{X}+(\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{b} \tag{4-75}$$

写成迭代形式得

$$\boldsymbol{X}_k=(\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{U}\boldsymbol{X}_{k-1}+(\boldsymbol{D}-\boldsymbol{L})^{-1}\boldsymbol{b} \tag{4-76}$$

式中，下标 $k=1, 2, 3, \cdots, n$。

这就是 Gauss – Seidel 迭代公式。把上述迭代公式编制成函数 M 文件如下：

```
% 线性方程组的 Gauss – Seidel 迭代解法
% A——系数矩阵
% b——常数向量
% X0——初值
% n——迭代次数
% X——解向量
function X = GS(A,b,X0)                    %函数定义项
D = diag(diag(A));                         %对角矩阵
L = - tril(A ,- 1);                        %下三角矩阵
U = - triu(A ,1);                          %上三角矩阵
B = (D - L) \ U;                           %迭代形式的系数矩阵
```

```
F = (D - L) \ b;                    %迭代形式的常数向量
X = B* X0 + F;                      %Jacobian 迭代公式
n = 1;
while norm(X - X0) > = 1e - 6       % 当 2 次解向量的长度大于万分之一时，
                                      继续迭代,否则结束迭代
    X0 = X;                         %上次解向量,赋值给下一个
    X = B* X0 + F; n = n + 1;       %迭代运算,迭代次数加 1
    if n > 500                      %若迭代次数大于 500 则中断
        break
    end
end
n                                   %显示迭代次数 n
```

关于迭代法能否收敛问题，它与初始向量 $\boldsymbol{X}_0$ 无关。主要取决于迭代矩阵 $\boldsymbol{B} = (\boldsymbol{D} - \boldsymbol{L}) \backslash \boldsymbol{U}$ 的特征值 λ_i。若 max（absλ_i）<1，则迭代收敛，否则将是发散的。有关证明，请查阅数学手册。

【例 4-20】 续上例矩阵 $\boldsymbol{A} = \begin{bmatrix} 3 & 2 & 1 \\ 3 & 4 & 3 \\ 1 & 2 & 5 \end{bmatrix}$，$\boldsymbol{b}$ = [2 3 4]，$\boldsymbol{X}_0$ = [0 0 0] 采用 G－S 迭代法，能否收敛，若能收敛，求其解。

解：

```
>>D = diag(diag(A))                 %提取对角线矩阵
D =

     3     0     0
     0     4     0
     0     0     5

>>L = - tril(A, - 1)                %提取下三角矩阵
L =

     0     0     0
    -3     0     0
    -1    -2     0

>>U = - triu(A,1)                   %提取上三角矩阵
U =

     0    -2    -1
     0     0    -3
     0     0     0

>>B = (D - L) \ U                   %构造迭代矩阵
B =
```

```
     0      -0.6667   -0.3333
     0       0.5000   -0.5000
     0      -0.0667    0.2667
≫eig(B)                                  %计算特征值,均小于 1,故收敛
ans =

         0
    0.6000
    0.1667
≫ X = GS(A,b,X0)                         %用 G-S 迭代法解方程组
n =                                      %迭代次数
    28
X =                                      %迭代解

     0.6000
    -0.3000
     0.8000
```

4.7 非线性方程组的解法

当方程组中的变量含有变量的乘积、变量是三角函数、变量是指数函数的方程组称为非线性方程组。非线性方程组没有一般的解法。通常采用迭代法、梯度法来求解。在 MATLAB 中可以选用函数 solve 或 fsolve 求解。函数 solve 在 4.4 节已作介绍，故不再赘述。函数 fsolve 的书写格式如下：

x = fsolve (fun, x0, options)

[x, fvaal] = fsolve (fun, x0, options)

式中，x、fval 为同维向量，x0 为向量 x 的初值。fun 为函数名，以函数 M 文件编写。

Options 为选项。选项的优化设置如下：

options = optimset('param1', valuel', param2', valuel, …)

式中的 “param” 有 29 项，详见 “help optimset”。对于其中常用的参数设置见表 4-3。

表 4-3 常用的参数设置

参　　数	值	说　　明
display	off | iter | final | notify	显示分级：不显示 | 显示迭代过程 | 最终显示 | 不收敛时作出报告
MaxFunEvals	正整数	允许函数的最大值
MaxIter	正整数	允许最大迭代次数
TolFun	正数	允许函数值的偏差
TolX	正数	允许 x 的偏差
Diagnostics	‘on’ | {‘off’}	输出关于函数趋向解的诊断信息 | 缺省时为关闭
DiffMaxChange	正数 | {‘1e-1’}	限制变量微商最大变化率 | 缺省时为 1E-1
DiffMinChange	正数 | {‘1e-8’}	限制变量微商最小变化率 | 缺省时为 1E-8

【例 4-21】 用符号变量求解下列非线性方程组：

$$\begin{cases} 2x_1 - x_2 - \exp(-x_1) = 0 & (4\text{-}77) \\ -x_1 + 3x_2 - \exp(-x_2) = 0 & (4\text{-}78) \end{cases}$$

解：

```
>> syms x1 x2                                  %设置符号变量
>> eq1 = 2* x1 - x2 - exp(- x1);               %输入方程式(4-77)
>> eq2 = - x1 + 3* x2 - exp(- x2);             %输入方程式(4-78)
>> [x1,x2] = solve(eq1,eq2)                    %解方程组,得 x1,x2 的字符解
x1 =
.49933616815828200392181351741690
x2 =
.39173890856870255050650195268821
>> x1 = eval(x1)                               %将字符 x1 转换成显示精度的数
x1 =
    0.4993
>> x2 = eval(x2)                               %将字符 x2 转换成显示精度的数
x2 =
    0.3917
>> eq1 = eval(eq1)                             %将字符 eq1 转换成显示精度的数
eq1 =
  1.1102e-016
>> eq2 = eval(eq2)                             %将字符 eq2 转换成显示精度的数
eq2 =
  -2.2204e-016
```

【例 4-22】 用非线性求解函数 fsolve 求解下列非线性方程组

$$\begin{cases} 2x_1 - x_2 - \exp(-x_1) = 0 & (4\text{-}79) \\ -2x_1 + 2x_2 - x_1 x_2 = 0 & (4\text{-}80) \end{cases}$$

解：

将式（4-79）、式（4-80）在 M 文件编辑器中，编制成函数 M 文件，文件名为 fune.m 如下：

```
function y = fune (x)
y = [2* x(1) - x(2) - exp(- x(1)); - x(1) + 2* x(2( - x(1)* x(2)];
```

在命令窗口输入如下程序：

```
>> x0 = [0;0];                                 %设置初值
>> options = optimset('display','iter');       %设置选项,显示迭代过程
>> [x,fval] = fsolve('fune',x0,options)        %解非线性方程组
                                    Norm of        First - order    Trust - region
Iteration     Func - count    f(x)     step           optimality       radius
```

```
0          3          1                         3          1
1          6          0.0113449     0.447214    0.0918     1
2          9          3.85927e-005  0.11547     0.00805    1.12
3          12         3.63398e-010  0.00706741  2.62e-005  1.12
4          15         2.74919e-020  2.12165e-005  2.3e-010 1.12
Optimization terminated successfully:                %最终优化成功
First-order optimality is less than options.TolFun.  %一阶优化后解的误差小于函数允差
x =                                                  %x 的解
    0.4656
    0.3034
fval =                                               %相应的函数值
  1.0e-009*
    -0.0223
    -0.1643
```

4.8 非负最小二乘解

对于超定方程组的解法，在 4.2 节已经提到过，它是采取最小二乘解。但是当问题附有非负约束时，采用线性最小二乘解就无能为力，而必须采非负最小二乘解 lsqnonneg。非负最小二乘解的书写格式为

```
x = lsqnonneg(C,d)
x = lsqnonneg(C,d,x0)
[x,resnorm] = lsqnonneg(…)
```

非负最小二乘解的表达式中 C 为系数矩阵，x0 为初值，d 为常数项向量，x 为解向量，resnorm 为剩余量的范数

resnorm = norm (C* x - d)

其中　　C* x = d

约束条件　　x > = 0

【例 4-23】　某工厂生产 A、B、C 三种产品，每种产品都经过三道工序，每道工序所需时间和可利用时间见表 4-4，问充分利用可利用时间的情况下，能生产多少产品。

表 4-4　某厂生产产品的时间表

工序 \ 产品	A/kg	B/kg	C/kg	可利用时间/h
P1/h	2	3	4	1200
P2/h	1	4	5	1800
P3/h	3	4	2	1000

解：

```
≫A = [2 3 4; 1 4 5;3 4 2]                    %3 种产品的工时消耗的矩阵
A  =
     2     3     4
     1     4     5
     3     4     2
≫b = [1500,2000,1600]';                       %线性方程组的常数项
≫x = inv(A)* b                                %解线性方程组
x =                                           %解答,A 产品为负,不合理
   -23.5294
    358.8235
    117.6471
≫x = lsqnonneg(A,b)                           %采用最小二乘法的非负解
x =
          0
    336.7347
    127.3469
≫x = fix(x)                                   %取整
x =
     0                                        %A 产品不生产
   336                                        %B 产品生产 336 千克
   127                                        %C 产品生产 127 千克
≫r = A* x - b                                 %工时消耗误差
r =
    16                                        %P1 工序有余
   -21                                        %P2 工序稍不足
    -2                                        %P3 工序稍不足
```

【例 4-24】 已知线性方程组

$$\begin{cases} 61x_1 + 40x_2 + 34x_3 = 605 \\ 79x_1 + 93x_2 + 32x_3 = 622 \\ 22x_1 + 91x_2 + 80x_3 = 819 \\ 80x_1 + 41x_2 + 55x_3 = 909 \\ 37x_1 + 89x_2 + 77x_3 = 866 \end{cases}$$

求非负解。

解： 从已知方程组得矩阵

```
≫ C = [61,40,34;79,93,32;22,91,80;80,41,55;37,89,77] %输入矩阵 C
C =
```

```
    61    40    34
    79    93    32
    22    91    80
    80    41    55
    37    89    77
>> d = [605;622;819;909;866];                    %输入向量 d
>> x1 = pinv(C)*d                                %最小二乘解
x1 =

    5.0000
   -1.0000
   10.0000

>> x = lsqnonneg(C,d)                            %非负最小二乘解
x =

    4.7026
         0
    9.0489
```

第 4 章习题

4-1　求解线性方程组 $\boldsymbol{AX}=\boldsymbol{b}$，其中

$$\boldsymbol{A}=\begin{bmatrix}5 & 8 & 1 & 0\\3 & 5 & 8 & 1\\0 & 3 & 5 & 8\\0 & 0 & 3 & 5\end{bmatrix},\quad \boldsymbol{b}=[34\ 65\ 90\ 50]。$$

4-2　求解下列超定方程组的最小二乘解

$$\begin{aligned}x_1+x_2&=5\\x_2+x_3&=6\\x_3+x_4&=7\\x_1+x_4&=8\\x_1+x_3&=9\\x_2+x_4&=10\end{aligned}$$

4-3　某数除 3 余 1、除 5 余 3、除 8 余 7、除 11 余 8，求此最小整数解。

4-4　求解二元方程组 $x+y=5$；$x^2+y^2=16$ 的解。

4-5　已知 3 点的坐标为[1,5]、[4,-3]和[-4,-1]，求解通过此 3 点的圆半径和圆心坐标。

4-6　用迭代法求解下列方程组。

(1) $\boldsymbol{A}=\begin{bmatrix}5 & 3 & 1 & 1\\2 & 5 & 3 & 1\\1 & 2 & 5 & 3\\1 & 1 & 2 & 5\end{bmatrix},\ \boldsymbol{b}=\begin{bmatrix}27\\40\\56\\54\end{bmatrix}$

(2) $\boldsymbol{A}_1=\begin{bmatrix}13 & 11 & 2 & 0 & 0\\ 7 & 15 & 11 & 2 & 0\\ 3 & 7 & 17 & 11 & 2\\ 0 & 3 & 7 & 19 & 11\\ 0 & 0 & 3 & 7 & 23\end{bmatrix}$, $\boldsymbol{b}_1=\begin{bmatrix}25\\ 30\\ 35\\ 40\\ 45\end{bmatrix}$

4-7 用非线性求解函数 fsolve 求解下列非线性方程组

$$\begin{cases}2x_1-x_2-\exp(-3x_1)=0\\ -x_1+2x_2-x_1x_2=0\end{cases}$$

4-8 已知线性方程组

$$\begin{cases}41x_1+40x_2+34x_3=600\\ 79x_1+93x_2+32x_3=650\\ 22x_1+91x_2+80x_3=800\\ 80x_1+41x_2+55x_3=900\\ 95x_1+89x_2+77x_3=1000\end{cases}$$

求非负最小二乘解。

第 5 章　数据的可视化

MATLAB 提供多种图形功能，它使你的数据或函数可视化，使数据不再是枯燥乏味的东西。使用 MATLAB 的图形函数，可以绘制二维或三维的数据图形和函数图形，如数据的散点图、直方图、茎干图、饼图、阶梯图和面积图等。在三维图形方面有三维线性图、网格图、彩色表面图、等高线图等。为了使数据可视化，基本步骤是：

(1) 准备好数据；

(2) 选择适用的绘制图形函数；

(3) 选择窗口和位置；

(4) 编辑图形标注和说明；

(5) 输出或保存图形。

用于数据可视化的常用命令见表 5-1：

表 5-1　数据可视化的常用命令

函　数　名	用法及说明
plot	二维线性图，书写格式为 plot（x，y，s），x、y 为向量，s 为线型字符串
plotyy	二维双纵坐标图，书写格式为 plot（x1，y1，x2，y2）
loglog	双对数坐标平面图
semilogx	半对数坐标平面图
axis	控制坐标轴刻度及坐标范围，书写格式为 axis（xmin，xmax，ymin，ymax）
axes	在任意位置建立坐标
figure	建立图形窗口
subplot	建立图形的平铺位置
hold on	图形保持
grid on	图形中添加栅格线
axis equal	设置坐标轴纵、横比率相等
axis square	设置当前纵坐标，横坐标轴为正方形
bar	条形图
area	面积图
pie	饼图
hist	直方图
rose	极坐标柱状图
stem	茎干图
stair	阶梯图
feather	羽毛图
quiver	箭形图
polt3	三维线性图
bar3	三维条形图

（续）

函数名	用法及说明
area3	三维面积图
pie3	三维饼图
contour	等高线图
mesh	三维网格图
Surf	三维曲面图

5.1 线性图函数 plot

线性图是绘制数据图形的基础，所以线性图函数 plot 在 MATLAB 中用得最广。它可以用来绘制散点图、序列图、向量图、矩阵图和函数图等。它可以设置线的类型（LineStyle），如实线、虚线、双点线、点划线，可以改变线的颜色（color）和线的宽度（LineWidth），也可以设置线图上的标记（marker）。还可以对图面进行标注，如增加标题、图例、坐标、文字标注等，使图形的内容更加丰富。它的书写格式如下：

```
plot(Y)
plot(X,Y)
plot(X1,Y1,LineStyle,X2,Y2,…)
plot(…,'PropertyName',PropertyValue,…)
h = plot(…)
```

式中，plot（Y）绘制以向量 Y 元素值为纵坐标（假如元素为实数），它的下标值为横坐标的线性图。假如 Y 的元素为复数，则以实数为纵坐标，虚数为横坐标绘制线性图。

对于格式 plot（X，Y），则以向量 X 为横坐标，Y 向量为纵坐标绘制线性图。若 X、Y 为同维矩阵，则以矩阵 X、Y 的对应列向量绘制线性图。

对于格式 plot（X1，Y1，LineStyle，…），则规定线的类型，如实线、虚线、点划线等。不作规定时，则默认为实线。

对于格式 plot（…，'PropertyName'，PropertyValue，…），则对特性名，特性值作规定。特性名有线宽（LineWidth）、颜色（Color）、标记点（Marker）、标记点边缘颜色（MarkerEdgeColor）、标记点充填颜色（MarkerFaceColor）、标记点大小（MarkerSize）等。下面举例说明它的应用。

对于格式 h = plot（…），则返回图形的句柄，它相当于图形的标识码。

对于线宽，以点宽数来表示，在默认的情况下线宽相当于 0.5 个点宽。1 个点宽，则相当于 1/72in 宽。对于默认情况下的标记点的直径为 5 个点宽，也可以用点宽数来设置标记点大小。

对于线型、标记点符号和颜色说明见表 5-2、表 5-3 和表 5-4：

表 5-2 线型（LineStyle）说明

线型	符号
实线	-
虚线	- -
双点线	:
点划线	- .

表 5-3　标记点（Marker）说明

标记点符号	说　　明	标记点符号	说明	标记点符号	说　　明
+	加号	s	方块	<	左三角
o	圆点	d	菱形	p	五角形
*	星号	^	上三角	h	六角形
.	点号	V	下三角		
x	叉号	>	右三角		

表 5-4　线的颜色（Color）说明

名　　称	缩　　写	RGB　　值	名　　称	缩　　写	RGB　　值
黄色	Y	[1 1 0]	绿色	g	[0 1 0]
紫红色	M	[1 0 1]	蓝色	b	[0 0 1]
兰绿色	C	[0 1 1]	白色	w	[1 1 1]
红色	R	[1 0 0]	黑色	b	[0 0 0]

现举例说明它的应用。

【例 5-1】　用均匀分布的随机函数，产生 1 行 8 列的向量，将该向量乘 100 后再沿零取整，得向量 Y，求向量 Y 的线性图 plot（Y）。

解：

```
≫Y = fix(100* rand(1,8))                %由均匀分布随机数，产生向量 Y
Y =
      45    93    47    41    84    52    20    67
≫plot(Y)                                %绘制 Y 向量的线性图，如图 5-1
```

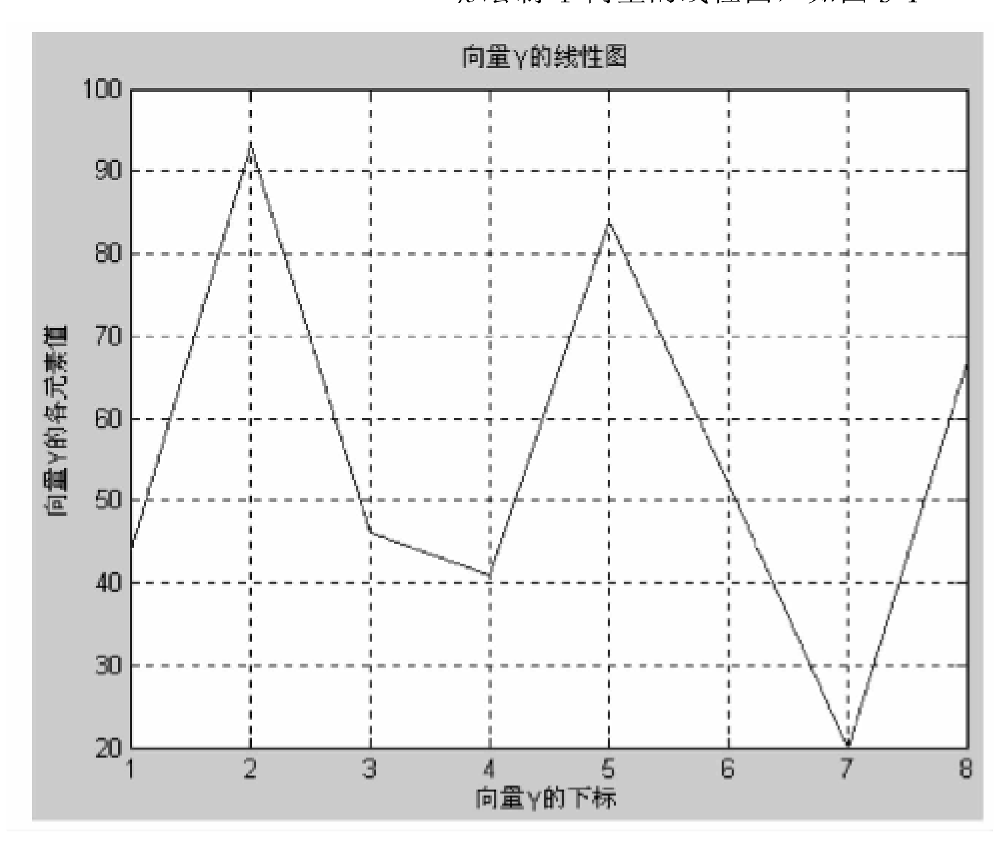


图 5-1　随机向量 Y 的线性图

```
>>grid on                              %增加坐标格栅线
>>title('向量 Y 的线性图')             %设置图形标题
>>xlabel('向量 Y 的下标')              %设置横坐标标注
>>ylabel('向量 Y 的各元素值')          %设置纵坐标标注
```

【例 5-2】 已知反正切函数 $y = \arctan x$，求其微分曲线并分别画出线性图。

解：

```
>>syms x                               %设置符号变量
>>y = atan(x);                         %建立反正切符号表达式
>>y1 = diff(y)                         %建立反正切函数的微分，diff 为微分函数，详
                                        见第 8 章

y1 =
1/(1 + x^2)
>>x = -5:0.1:5;                        %设置自变量 x 向量
>>y = eval(y);                         %执行反正切函数的计算
>>y1 = 1./(1 + x.^2);                  %计算微分曲线的值
>>plot(x,y,'-',x,y1,'*')               %绘制反正切函数及微分曲线，如图 5-2 所示
>>grid on                              %增加坐标格栅线
>>title('反正切函数及其微分曲线')      %设置标题
>>xlabel('x 坐标')                     %设置横坐标
>>ylabel ('y 坐标')                    %设置纵坐标
```

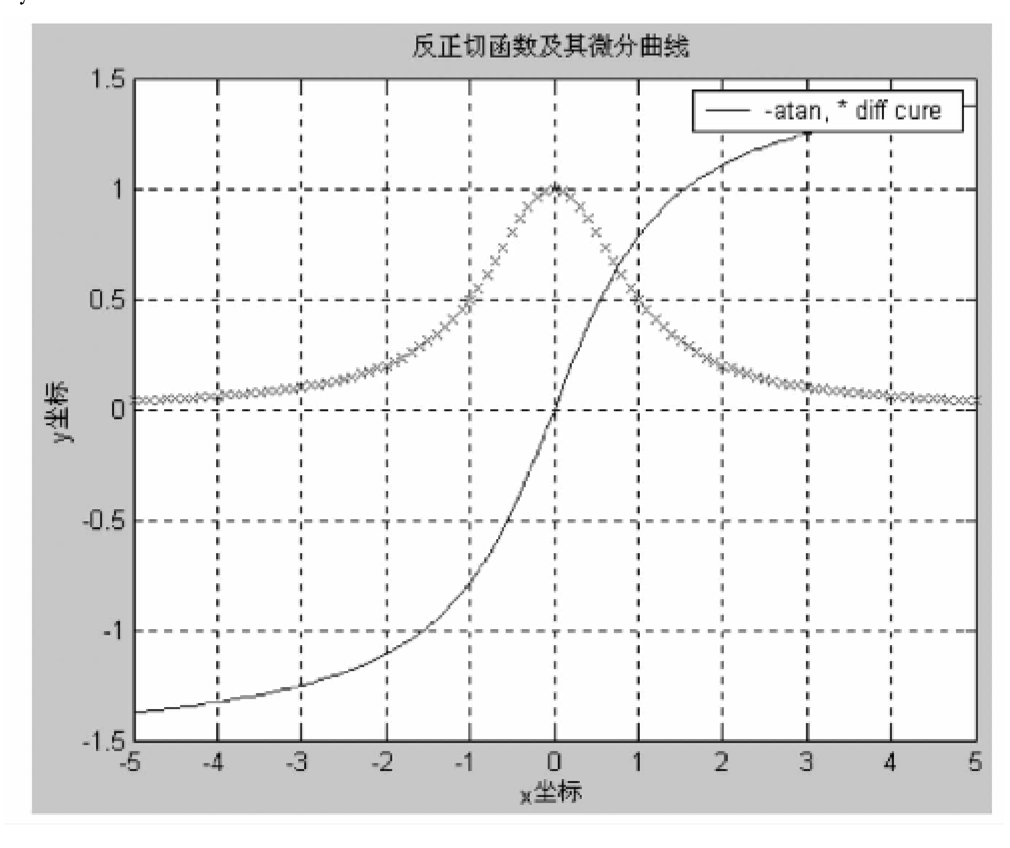


图 5-2 反正切函数及其微分曲线

≫legend('－atan','* diff curve') %设置图例，实线显示反正切函数，* 号线显示反正切函数的微分曲线

【例 5-3】 已知三角函数 $y=\tan(\sin x)$ 及三角函数 $y_1=\sin(\tan x)$，请绘出在区间 $[-\pi,\pi]$ 的 y、y_1 的线性图。

解：

在命令窗口输入如下程序：

```
≫x = - pi: pi/10: pi;                      %设置自变量向量 x
≫y = tan(sin(x));                          %设置函数向量 y，它与向量 x 必须是同维
≫y1 = sin(tan(x));                         %设置函数向量 y1，它也应该与向量 x 同维
≫plot(x, y, '- - rs', 'linewidth', 2, 'markeredgecolor', 'k', 'markerfacecolor', 'g', 'markersize', 10)
                                            %绘制 x，y 的线性图，线型为红色虚线，设置线宽为 2，标记点为方块，边缘线为黑色，设置标记表面为绿色，标记大小为 10。如图 5-3 所示
≫hold on                                    %图形保持
```

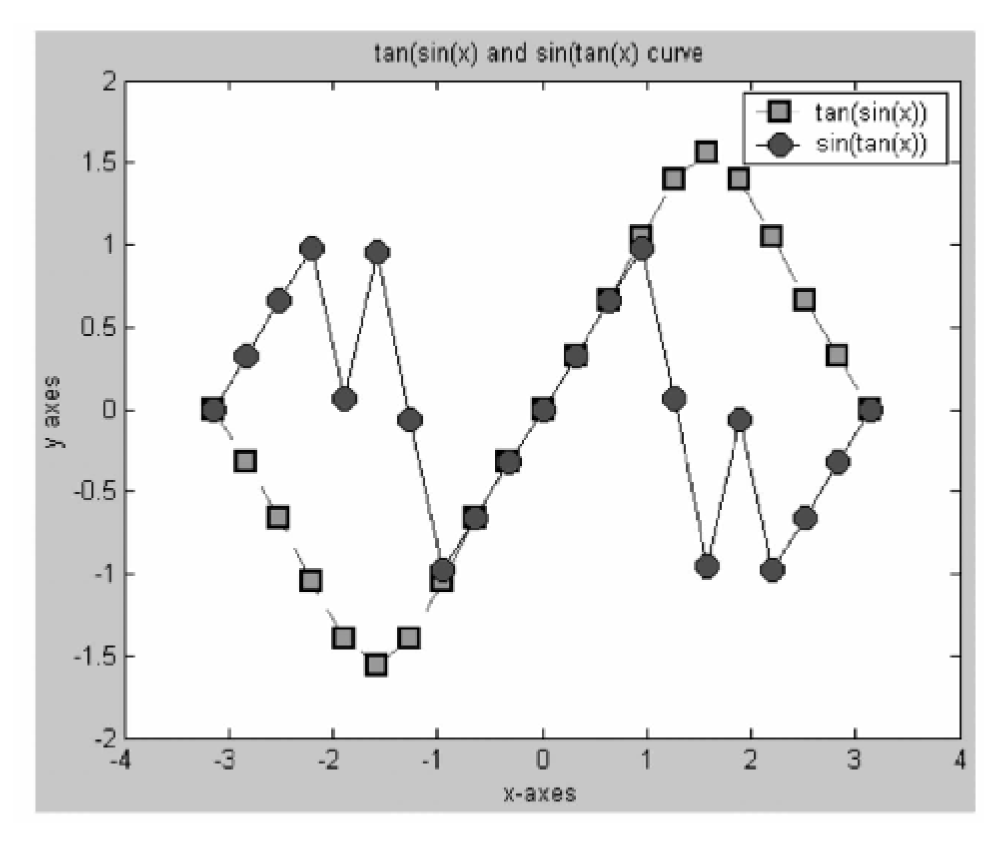


图 5-3 tan（sin x），sin（tan x） 图

```
≫plot(x, y1, '- ko', 'markeredgecolor', 'k', 'markerfacecolor', 'r', 'markersize', 10)
                                            %绘制 x，y1 的线性图，线型为黑色实线，标记点为圆点，标记边缘为黑色，标记表面为红色，标记点大小为 10)
```

```
>>xlabel('x - axes')                      %坐标轴标注
>>ylabel('y - axes')
```

【例 5-4】 已知 $y=\sin x$，绘制 [0，2π] 的函数图，并求函数图的句柄；通过 set 命令改变线宽为 8，线的颜色为红色。

解：

```
>>x = 0:pi/50:2*pi;                       %自变量取值范围
>>y = sin(x);                             %正弦函数
>>h = plot(x,y)                           %绘制二维线性图
h =                                       %函数图句柄值
    3.0016
>>set(h,'linewidth',8,'color','r')        %设置线宽和颜色，如图 5-4 所示
>>grid on                                 %增加坐标轴格栅线
```

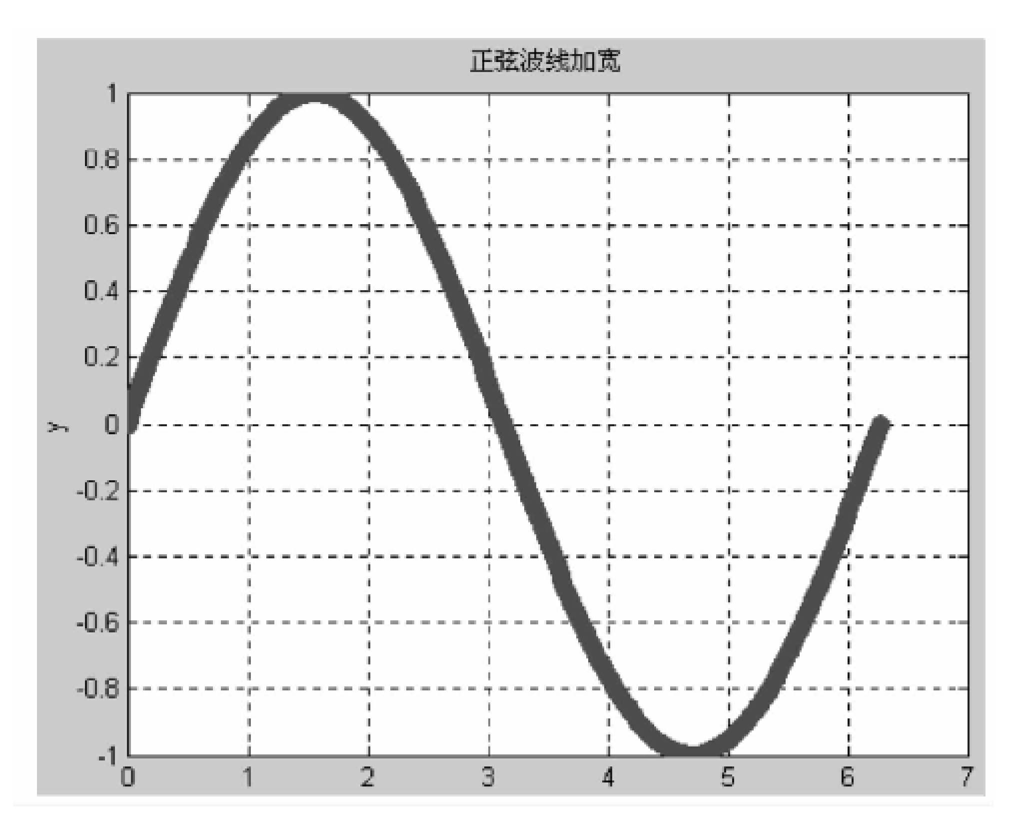


图 5-4　正弦波曲线的线加宽

【例 5-5】 已知以下单相电流、电压方程式如下，求功率曲线，并图解之。

$I=\sin t$

$V=2\sin(t+\pi/3)$

$P=IV=2\sin t\sin(t+\pi/3)$

解：

```
>>t = 0:pi/50:4*pi;                       %设置时间变量 t，变化范围
>>I = sin(t);                             %电流向量
>>V = 2*sin(t + pi/3);                    %电压向量
```

```
≫P = I.* V;                                 %功率向量，主电流向量与电压向量的数组相乘
≫plot(t,I,'-',t,V,':',t,P,'+');grid on      % 绘制 I、V、P 曲线，分别以实线，双点线，加号线表示，并显示坐标栅格线，如图 5-5 所示
```

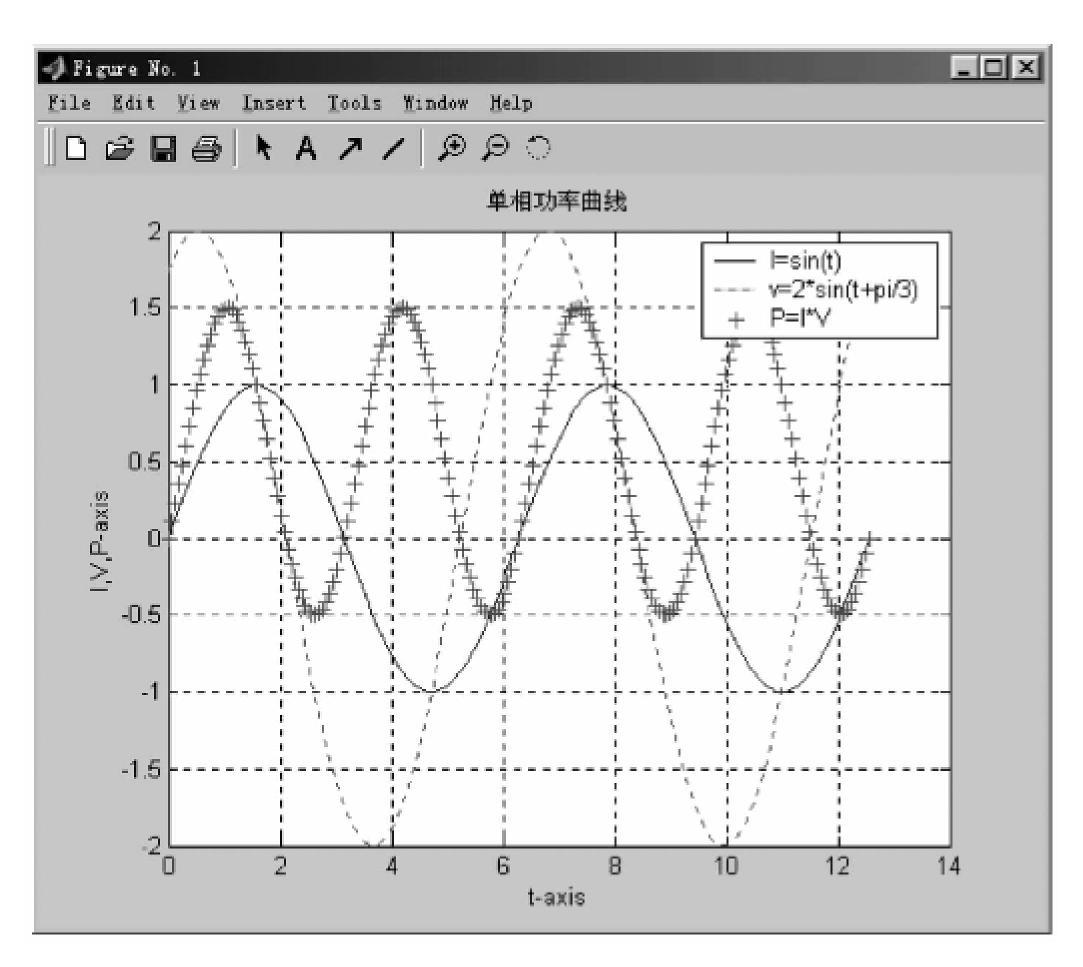


图 5-5 单相功率曲线

【例 5-6】 已知 $u(t)=0$，　当 $t<0$；

$u(t)=1$，　当 $0\leqslant t<2$；

$u(t)=0.2$，　当 $t>=2$。求 $u(t)$ 在 $[0,10]$ 范围内的线性图。

解：

```
≫time = [0:0.02:10]';                       %设时间向量
≫u = 1.0*(1+0*(time));                      %设置 u = 1.0 的直线。
≫for i = min(find(time> =2.0)):length(u)    %在 time>2.0 时，u(t) = 0.2
u(i) = 0.2;
end
≫plot(time,u)                               %绘制线性图，如图 5-6 所示
≫axis([0,10,0,1.2])                         %设置坐标范围
≫title('脉冲波')                             %图标题
```

```
>>xlabel('time')                              %坐标轴标签
>>ylabel('u')
```

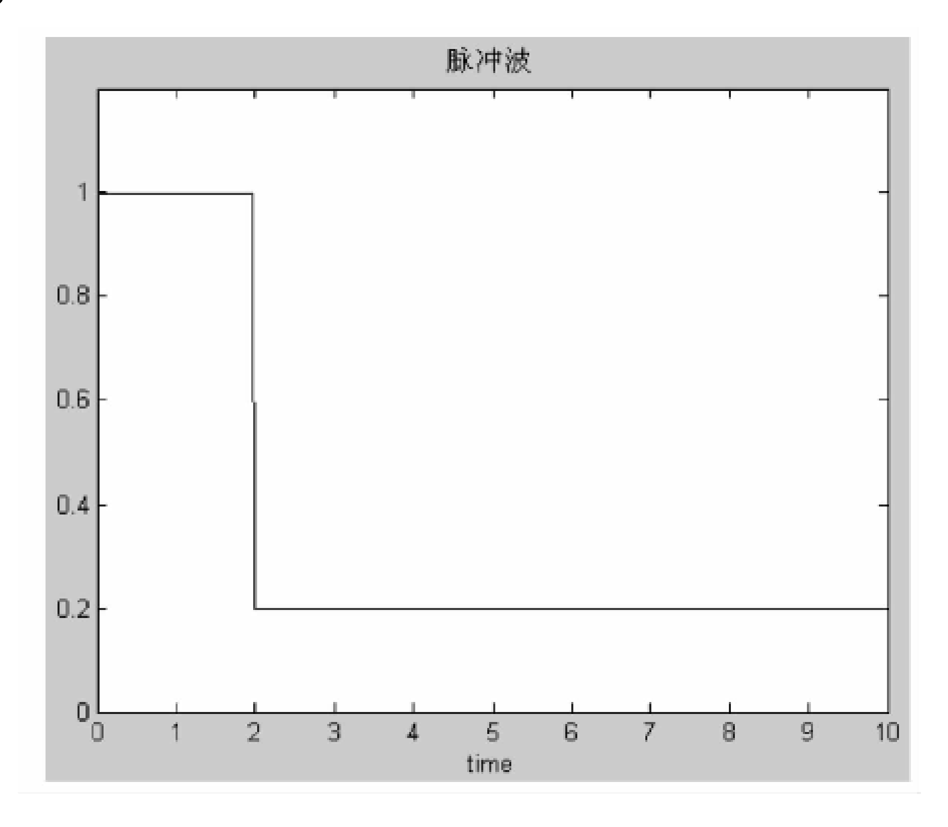


图 5-6 脉冲波的绘制

5.2 简易线性函数图

为了绘制线性函数图的方便，而设置的简易线性图函数 ezplot，它不用设置自变量的间隔向量、线宽、标记点、颜色，只要知道函数的符号表达式，即可绘出函数图形。ezplot 的书写格式为

```
ezplot(f)
ezplot(f,[xmin,xmax])
ezplot(f,[xmin,xmax,ymin,ymax])
ezplot(f(x,y))
ezplot(f(x,y),[tmin,tmax])
```

式中，f 为符号函数表达式，默认的自变量变化区间为 $[-2\pi, 2\pi]$，亦可自行选择自变量变化区间为[xmin,xmax]。对于隐函数 $f(x,y)=0$，在默认的情况下变量 x，y 的变化区间均为 $[-2\pi, 2\pi]$，亦可自行选择自变量变化区间为[xmin,xmax,ymin,ymax]。对于变量 x，y 均为参变量 t 的函数，则可在书写格式中添加[tmin,tmax]。

【例 5-7】 已知椭圆方程式 $x^2/9+y^2/4=1$，求绘制椭圆曲线。

解：

```
>>ezplot('x^2/9+y^2/4-1')                     %绘制椭圆曲线，图形从略
>>axis([-3,3,-2,2]),grid on                   %设置坐标轴范围，添加栅格线
```

【例 5-8】 已知三角函数 $y=\cos(2x)/(1-\sin 2x)^{1/2}$，求函数图形。

解：

```
≫ezplot('cos(2*x)./(1-sin(2*x)^(1/2))')          %绘制函数图形，如图 5-7 所示
≫grid on
```

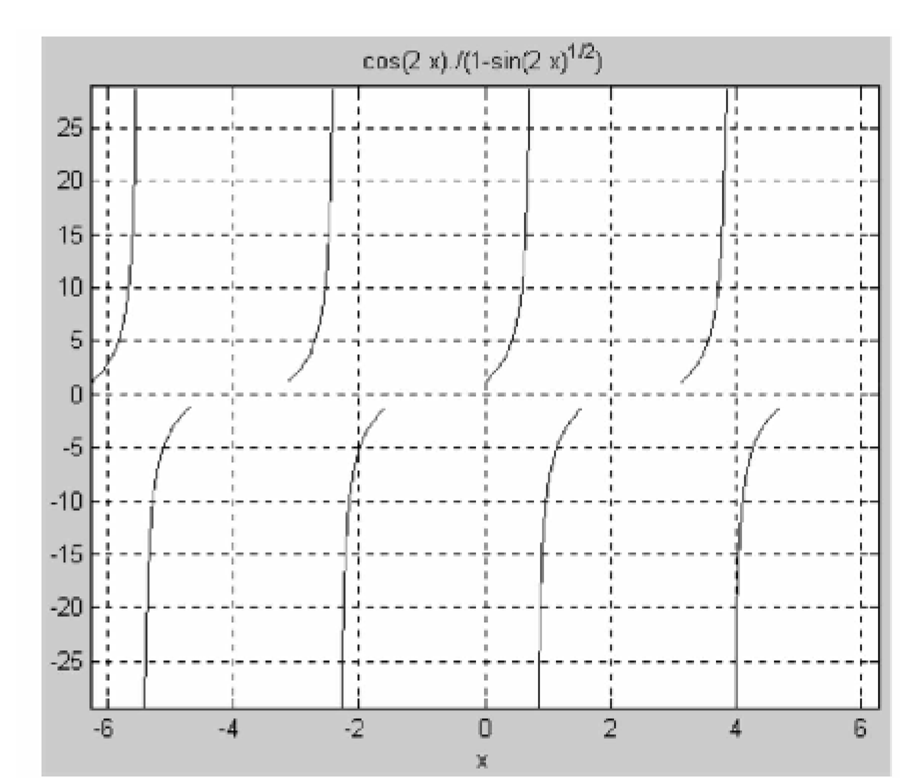


图 5-7　用 ezplot 绘制 $y = \cos 2x / (1 - 2\sin 2x)^{1/2}$ 的曲线图

5.3　散点图

在统计分析中，常常需要用坐标纸画散点，以便寻找数据的统计规律。使用 MATLAB，就可以不必手工画点了，只需要列出相关数据的数据向量，用 MATLAB 中的绘制线性图函数 plot 来处理了。另一种是使用散点图函数 scatter，其书写格式为

```
scatter(X,Y,S,C)
scatter(X,Y)
scatter(…,markertype)
scatter(…'filled')
h = scatter(…)
```

式中，X，Y 为横坐标向量和纵坐标向量，它们必须有相同的长度。S 是指圆圈标记点的面积，面积定为点宽的平方，它可以是标量，若是向量，则必须与 X、Y 长度相同。C 确定标记点的颜色，若是向量，则必须与 X、Y 具有相同的大小。markertype 用来改变标记点的类型，filled 用来充填标记点的颜色。在标记类型默认的情况下，scatter 绘出的是圆圈图。h 用来取得散点图的句柄，用于图形的修饰。

下面举例予以说明。

【例 5-9】　为了统计在正常情况下，人的体重与身高的关系，今有以下测试数据，见表 5-5，请画出它的散点图。并用线性回归找出它的回归方程。

表 5-5　被测试的 12 个人的身高与体重

序　　号	1	2	3	4	5	6	7	8	9	10	11	12
体重/kg	60	57	56.5	65	63	64	70	65	68	76	72	78
身高/m	1.55	1.58	1.60	1.64	1.66	1.68	1.70	1.73	1.78	1.80	1.82	1.85

解：

```
>>X = [1.55,1.58,1.6,1.64,1.66,1.68,1.7,1.73,1.78,1.8,1.82,1.85];
                                          %输入身高向量
>>Y = [60,57,57,65,63,64,70,65,68,76,72,78];  %输入体重向量
>>plot(X,Y,'b*')                          %绘制散点图，星号为标记点
>>hold on                                 %图形保持
>>polyfit(X,Y,1)                          %取 1 阶线性回归（详见第 6 章）
ans =
  63.2899 -41.2900
>>x = 1.5:0.1:2;                          %设 x 向量
>>y = -41.29 + 63.2899*x;                 %散点的回归直线
>>plot(x,y,'r-')                          %绘制回归直线，如图 5-8 所示
>>grid on                                 %增加坐标栅格线
>>xlabel('身高')                          %x 轴标注
>>ylabel('体重')                          %y 轴标注
>>title('身高与体重的关系曲线')           %图标题
```

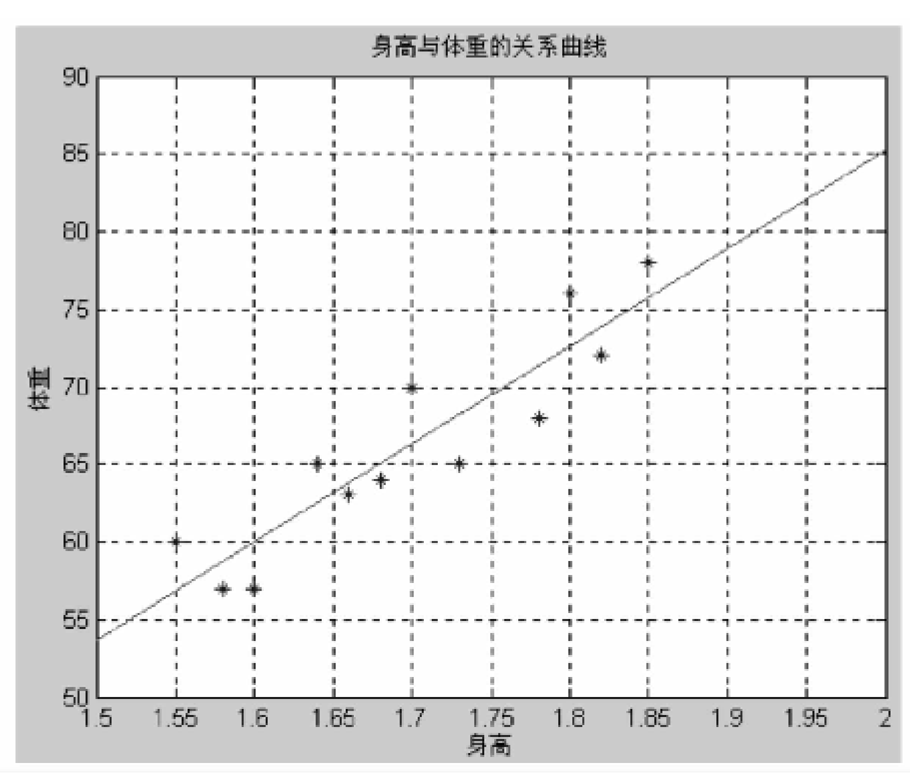


图 5-8　身高与体重的散点图及其线性回归线

【例 5-10】　有一台电气设备，从投入使用一直到报废，故障记录见表 5-6，请画出故障

曲线并用曲线加以平滑连接。

表 5-6　某电气设备故障记录

时间	1~3月	4~6月	7~9月	10~12月	2年	3年	4年	5年	6年	7年1~6月	7年7~12月
故障次数	4	3	2	1	0	0	1	0	1	3	6

解：

```
>>A=[0.25,0.5,0.75,1,2,3,4,5,6,6.5,7;4,3,2,1,0,0,1,0,1,3,6]
                                                    %由故障表输入的故障次数矩阵
A=
  Columns 1 through 8
      0.2500    0.5000    0.7500    1.0000    2.0000    3.0000    4.0000
      5.0000
      4.0000    3.0000    2.0000    1.0000         0         0    1.0000
           0
  Columns 9 through 11
      6.0000    6.5000    7.0000
      1.0000    3.0000    6.0000

>>t=A(1,1:11);                                      %从矩阵A提取故障时间向量
>>faunum=A(2,1:11);                                 %从矩阵A提取故障次数向量
>>t1=0:0.1:7;                                       %设置平滑故障曲线的时间向量
>>fauspl=interp1(t,faunum,t1,'spline');             %用3次样条插值联结故障点，
                                                     有关插值分析详见第6章
>>plot(t,faunum,'o',t1,fauspl)                      %绘制故障次数散点图和样条插
                                                     值图如图5-9所示
```

【例 5-11】　由国家计划生育委会公布的，到 2050 年我国人口增长的模型见表 5-7，请绘制增长曲线。

表 5-7　我国人口增长模型

年　　份	2001	2005	2010	2020	2050
人口/亿	12.76	13.3	14	15	16

解：

```
>>A=[2001 2005 2010 2020 2050;12.76 13.3 14 15 16]  %输入人口增长模型矩阵
A=
  1.0e+003*
    2.0010    2.0050    2.0100    2.0200    2.0500
    0.0128    0.0133    0.0140    0.0150    0.0160
>>year=A(1,:)                                       %从矩阵A中提取年份向量
year=
        2001      2005      2010      2020      2050
```

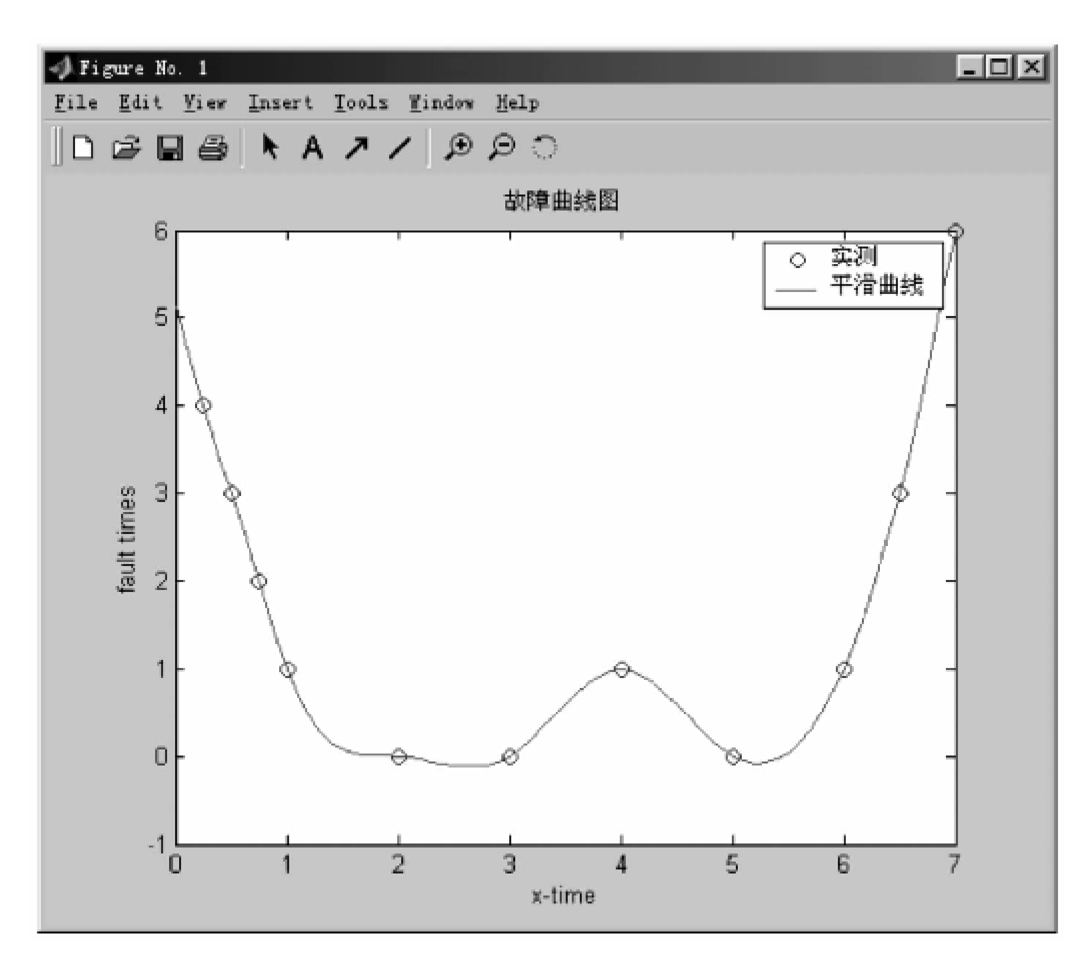


图 5-9　故障曲线统计图

```
≫popu = A(2,:)                                  %从矩阵 A 中提取人口向量
popu =
      12.7600     13.3000     14.0000     15.0000     16.0000
≫t = 2001:0.2:2050;                             %设时间向量，步长为 0.2 年
≫fspl = interp1(year,popu,t,'spline');          % 设置样条插值，interp1 为插
                                                   值函数，详见第 6 章
≫plot(year,popu,'o',t,fspl);grid on             %绘制曲线图，如图 5-10
```

【例 5-12】　利用散点图函数，设计花卉图。

解：

```
% This program for draw a scatter flower        %程序标题
for r = 2:20                                    %设置花卉直径从 2-20
    zeta = 0:pi/12:2* pi;                       %设置将圆 20 等分
    x = r* cos(zeta + r* pi/30);                %计算横坐标向量
    y = r* sin(zeta + r* pi/30);                %计算纵坐标向量
    c(1) = 2/r;c(2) = r/20;c(3) = r/20;         %设置颜色向量
    scatter(x,y,5,c,'filled')                   %绘制散点图
    hold on                                     %保持图形
end                                             %循环
```

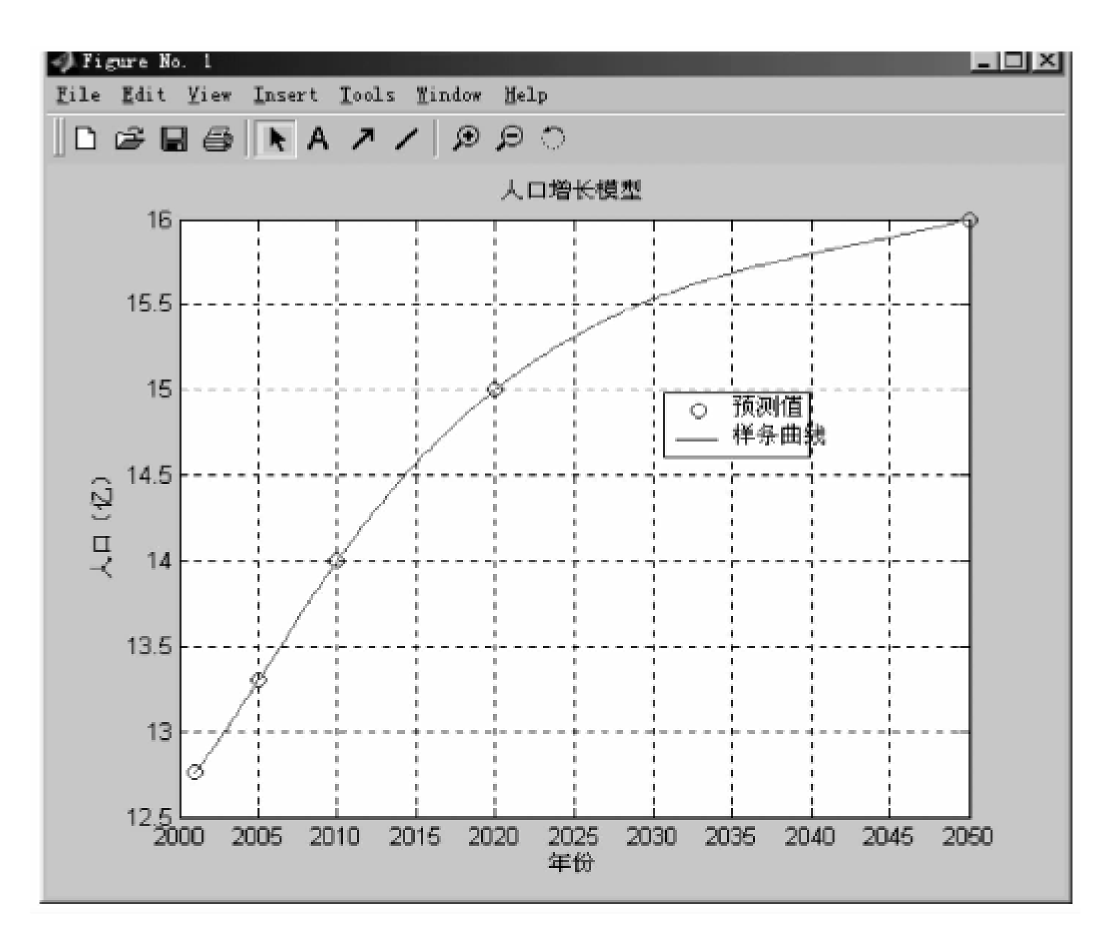


图 5-10 到 2050 年人口增长模型

运行该程序，程序名为 scatter _ flowerp 得散点图图形如图 5-11，图形文件名为 scatter _ flower. fig。

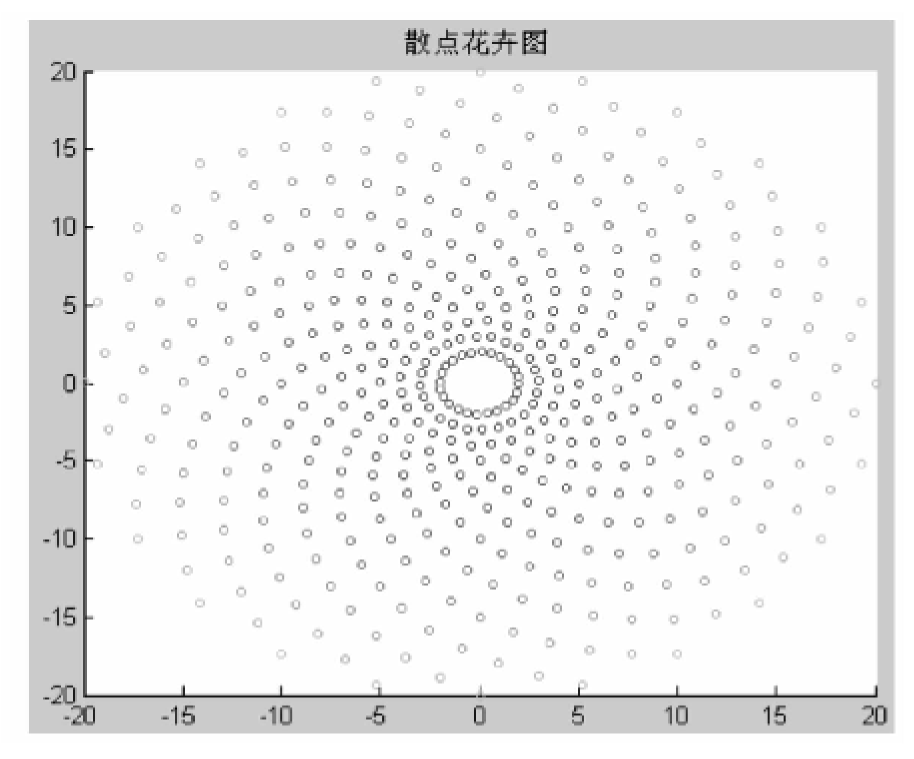


图 5-11 散点花卉图

5.4 极坐标图及其与直角坐标图的转换

有的函数曲线用直角坐标表示比较复杂，但如果用极坐标表示则比较简单，亦有相反的情况。MATLAB 既可以用直角坐标画图亦可以用极坐标画图，还可以互相转换。极坐标绘图函数的书写格式如下：

polar(theta,rho)

polar(theta,rho,LineSpec)

式中，theta 为模向量与横坐标的夹角向量，以弧度表示；rho 为模长度的向量；LineSpec 为线的规格与 5.1 节所述相同。

极坐标与直角坐标相互转换的关系如下：

极坐标转换成直角坐标的书写格式为

[x,y]=pol2cart(theta,rho)

直角坐标转换极坐标的书写格式如下：

[theta,rho]=cart2pol(x,y)

式中，x、y 分别为横坐标向量和纵坐标的向量。下面举例予以说明。

【例 5-13】 试用极坐标和直角坐标，分别绘制 3 叶玫瑰线 $r=\sin$ (3*zeta)。

解：

```
>>zeta=0:6*pi/600:6*pi;                  %设置角度向量
>>r=sin(3*zeta);                         %3 叶玫瑰线向量
>>[x,y]=pol2cart(zeta,r);                %将极坐标转换成直角坐标，pol2cart 为转换指令
>>subplot(1,2,1);polar(zeta,r)           %设置绘图窗口成 1 行 2 列，在左侧绘制极坐标形式的 3 叶玫瑰线
>>subplot(1,2,2);plot(x,y);grid on       %设置绘图窗口，在右侧绘制直角坐标形式的 3 叶玫瑰线，添加坐标格栅线。图形如图 5-12 所示
```

【例 5-14】 已知直角坐标参数方程 $x=\cos^3$(zeta)，$y=\sin^3$(zeta)，将其转换成极坐标，并图示之。

解：

```
>>t=0:pi/20:4*pi                         %设置参变量 zeta 的变化范围
>>x=cos(zeta).^3;                        %参数方程式，x 向量
>>y=sin(zeta).^3;                        %参数方程式，y 向量
>>[theta,rho]=cart2pol(x,y)              %将直角坐标转换成极坐标，theta 为幅角，rho 为幅值
>>h=polar(theta,rho)                     %绘制极坐标图，如图 5-13 所示
h=                                       %极坐标图的句柄
  227.0005
>>set(h,'linewidth',2)                   %设置线宽为 2
```

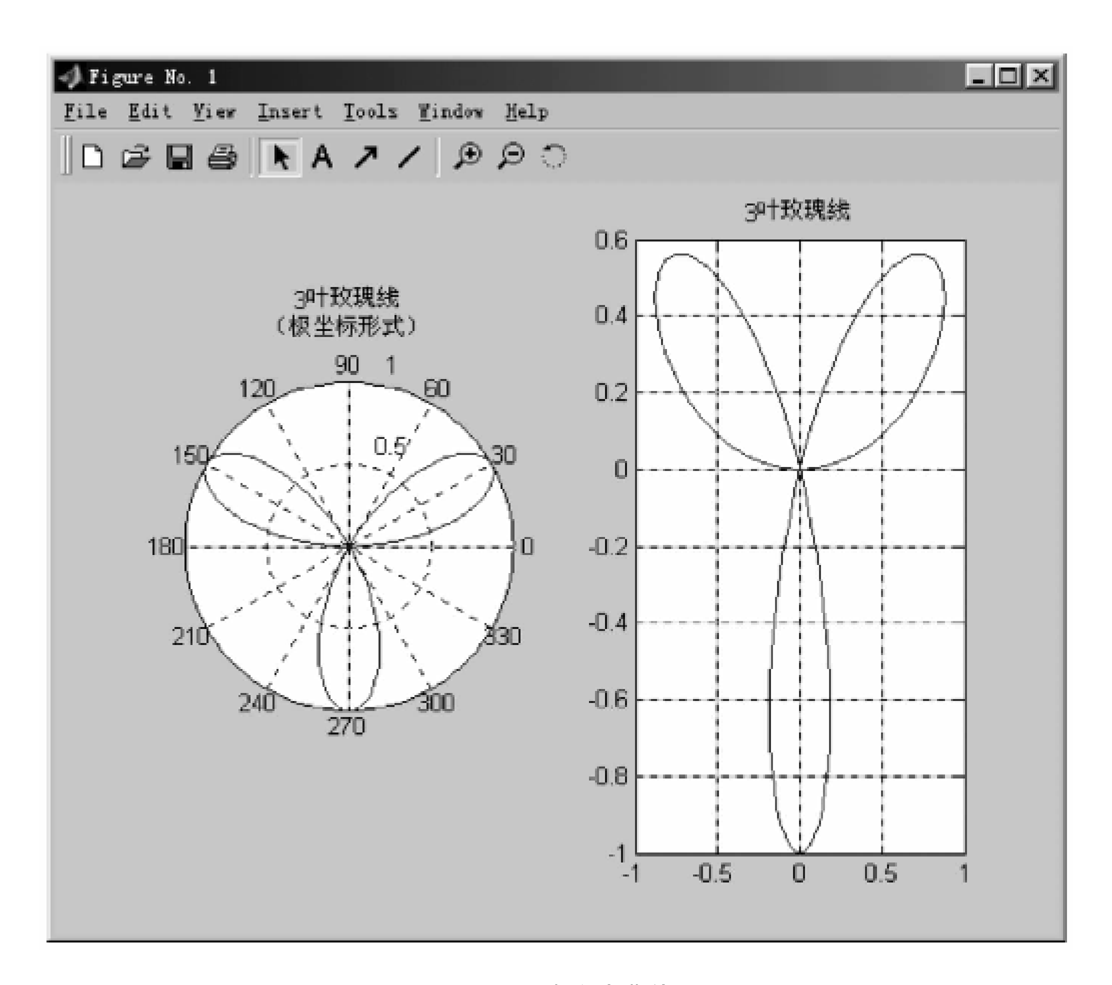


图 5-12　3 叶玫瑰曲线图

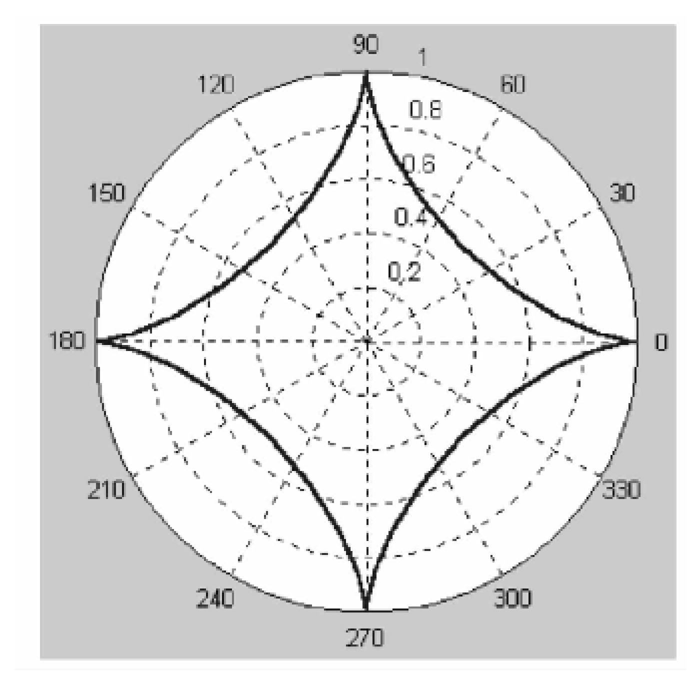


图 5-13　内摆线，从直角坐标转换成极坐标

5.5　条形图

条形图是用来显示向量或矩阵的元素值在水平或者垂直方向。条形图也是形像地显示数

据的工具。条形图函数的书写格式如下：

```
bar(Y)
bar(x,Y)
bar(…,width)
bar(…,'style')
bar(…,LineSpec)
h = bar(…)
barh(…)
```

bar(Y)绘制向量 Y 每一个元素的条形图，条形的幅值代表元素值，条形图的下标为向量 Y 的序列。假若 Y 为矩阵，则产生代表矩阵每一行元素值的条形组，条形组的横坐标是显示矩阵的行数。

对于bar(x,Y)，则绘制向量 Y 的每一个元素在指定 x 位置的条形图，向量 x 的值必须是单调递增的。假若 Y 为矩阵，则一串条形代表矩阵 Y 的行元素，放置于横坐标 x 的位置。

对于bar(…,width)用来设置条形的宽度，默认时为 0.8，当宽度设为 1 时，则条形一个紧挨着一个，没有间隙。若没有指定 x，则条形组之间有细小的分离。

对于（…,'style'）用来指定条形的类型，类型分为“grouped”或“stacked”，“group”是默认的显示模式。“grouped”表示显示 n 组 m 个垂直条形图，n 表示矩阵 Y 的行数，m 则表示矩阵 Y 的列数。“stacked”则用元素叠加形式显示条形，它的高度是每行元素的总和，每行的条形是多色的，用颜色来区分各元素及其所占成分。

对于bar(…,LineSpec)用来设置条形的颜色。

对于h = bar(…)则返回句柄向量，为图形对象修饰之用。

对于barh(…)则刨建水平方向绘制的条形图。

【例 5-15】 绘制衰减余弦曲线 $y = \exp(0.5x)\cos x$ 在 $[0, 5\pi/2]$ 区间的条形图。

解：

```
≫x = 0:pi/20:5/2*pi;              %设置向量 x
≫y = exp(-0.5*x).*cos(x);         %计算向量 y
≫bar(y,0.5)                       %绘制条形图，宽度设为 0.5，图形如图 5-14 所示
```

【例 5-16】 用 group 和 stack 条形图分别显示 4 阶魔方矩阵。

解：

```
≫Y = magic(4)                     %输入魔方矩阵
Y =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

≫subplot(1,2,1)                   %设置 1 行 2 列的子图的第一图
≫bar(Y)                           %绘制以行为组的条形图
≫subplot(1,2,2)                   %设置 1 行 2 列的子图的第二图
```

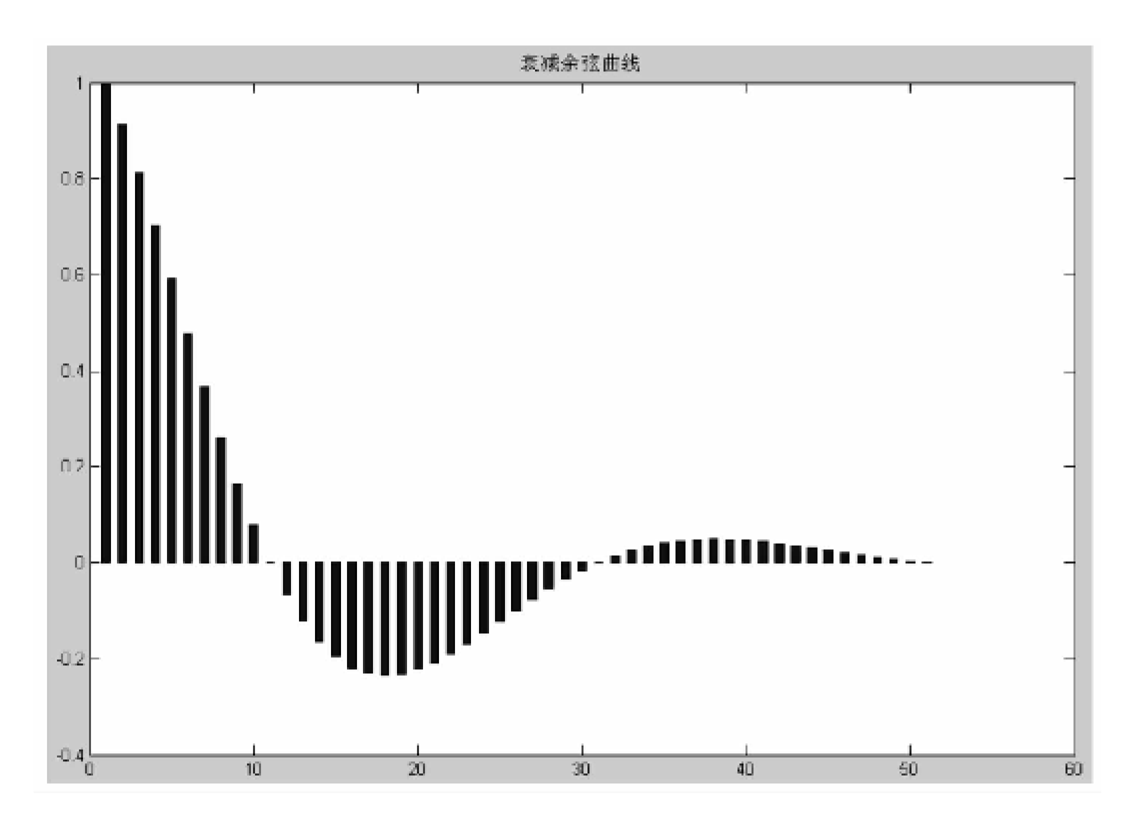


图 5-14　衰减余弦曲线的条形图

```
>>bar(Y,'stacked')                    %绘制以行元素叠加形式的条形图，如图 5-15 所示
```

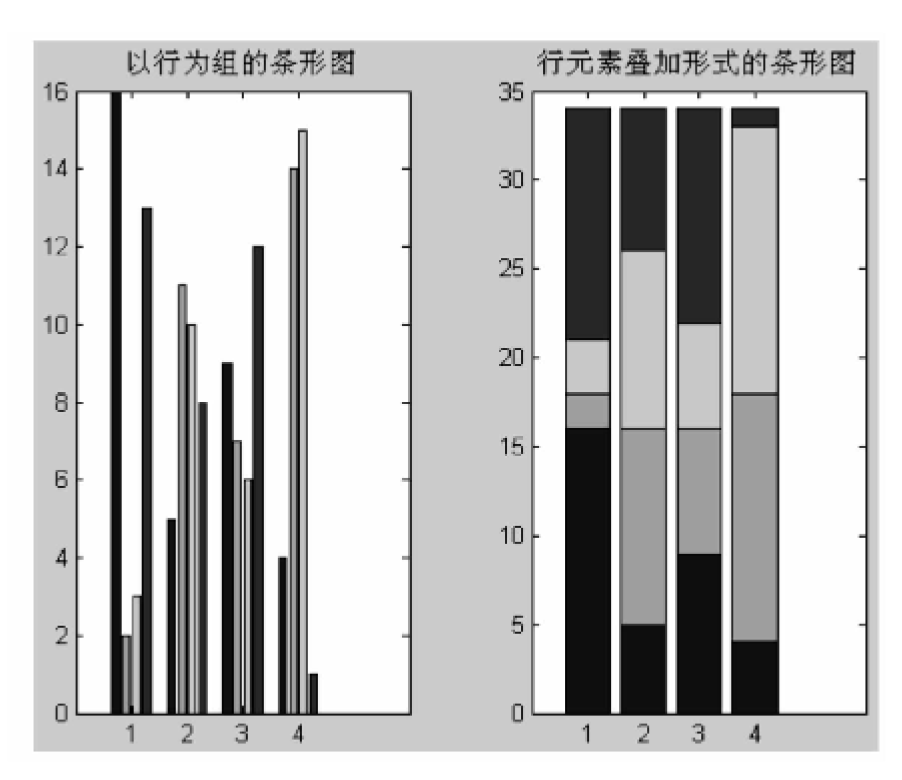


图 5-15　魔方矩阵的条形图

【例 5-17】　已知 4 阶 pascal 矩阵 Y，求其水平条形图及叠加模式水平条形图。

解：

```
>>Y = pascal(4)                    %输入 4 阶 pascal 矩阵
Y =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20

>>subplot(2,1,1)                   %设置 2 行 1 列的子图的第一图
>>barh(Y)                          %绘制水平条形图
>>subplot(2,1,2)                   %设置 2 行 1 列的子图的第二图
>>barh(Y,'stacked')                %绘制叠模式的水平条形图如图 5-16
```

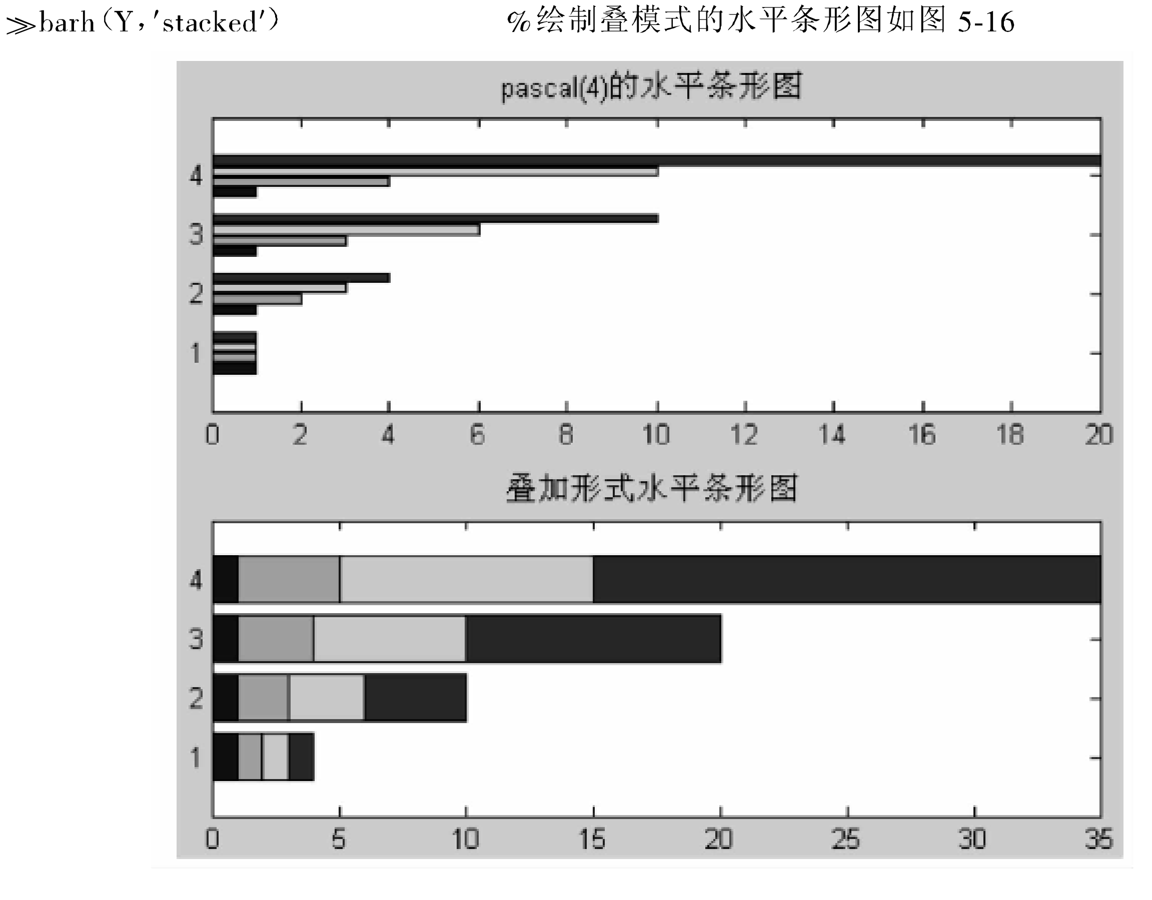


图 5-16　4 阶 Pascal 矩阵的水平条形图

【例 5-18】　今有三家企业 A、B、C 从 2001 年到 2004 年的销售量见表 5-8。

表 5-8　三家企业 2001 ~ 2004 年销售量

年　份	企　业　A	企　业　B	企　业　C
2001	105	56	107
2002	108	78	95
2003	110	105	73
2004	121	175	45

绘制条形图，比较三企业的优劣。

解：

```
≫A = [105,56,107;108,78,95;110,105,73;121,175,45]     %输入销售矩阵，第 1 列表示企业
                                                       A，第 2 列表示企业 B，第 3 列表
                                                       示企业 C
A =

     105     56     107
     108     78      95
     110    105      73
     121    175      45

≫x = [2001,2002,2003,2004];                            %输入年份向量
≫bar(x,A)                                              %绘制条形图
≫hold on                                               %保持图形
≫plot(x,A(:,2),'r-','linewidth',5)                     % 绘制企业 B 的增长曲线，线型为
                                                        红实线，线宽为 5，图形如图5-17
                                                        山图可知企业 B 为优，企业 A 次
                                                        之，企业 C 较差
```

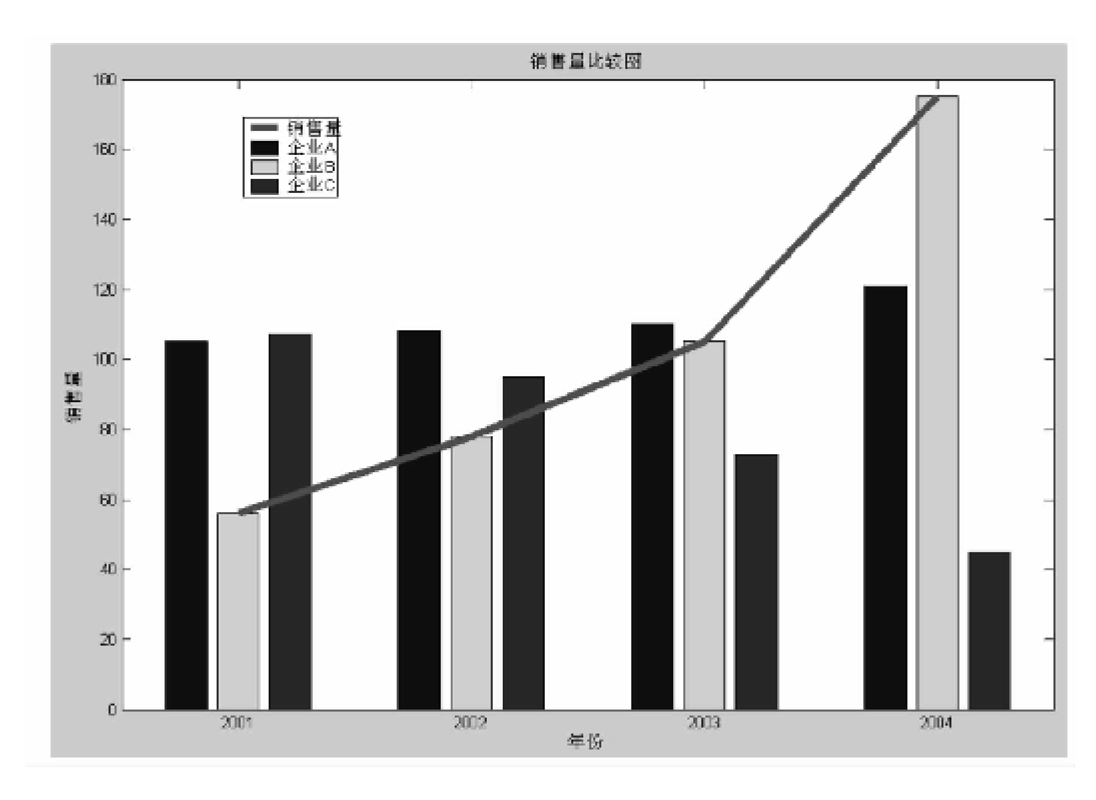


图 5-17　A、B、C 企业从 2001～2004 年销售量比较图

5.6　饼图

描述一组数据中每一分量占总体的份额或百分数，并用圆饼的一部分的扇形来表示，称为饼图。它是常用的，并且形象地显示出数据各个分量所占百分比的图。在 MATLAB 中饼图的书写格式为

pie(X)
pie(X,explode)
h = pie(…)

式中，X 为向量，pie 为绘制饼图的函数。pie(X)则绘出向量 X 中的每一分量；pie(X，explode)则将向量中某几个元素从饼图中分离出来。explode 必须是与 X 同维向量，它是一个以 0 或非零为元素的向量，如果explode(j)为非零；则X(j)元素从饼图中分离开来，否则保留。下面举例予以说明。

【例 5-19】 某公司销售 4 种产品 A、B、C、D，其每季对利润的贡献见表 5-9，求哪种产品对公司利润贡献最大，并从饼图中分离出来。

表 5-9 某公司销售的产品对利润的贡献

季　度	产　品　A	产　品　B	产　品　C	产　品　D
1	22	20	33	41
2	35	19	74	84
3	81	19	74	52
4	13	19	46	67

解:

```
>>A = [22 20 33 41;35 19 74 84;81 60 44 52;13 19 46 67]  %输入利润矩阵
A =
    22    20    33    41
    35    19    74    84
    81    60    44    52
    13    19    46    67
>>S = sum(A)          % 计算 4 种产品的年贡献，sum 为列向求和函数
S =
    151    118    197    244
>>[m,i] = max(S)      % 寻找最大值为 244，为产品 D。max 为寻找最大值函数，m 为最大值，i 为最大值的下标
m =
    244
i =
      4
>>explode = zeros(size(S))    %构造分离向量 expolde
explode =
    0    0    0    0
>>explode(i) = 1      %设置分离产品 D
```

```
explode =
     0     0     0     1
≫pie(S,explode)                         %绘制饼图，并分离产品D，如图5-18所示
```

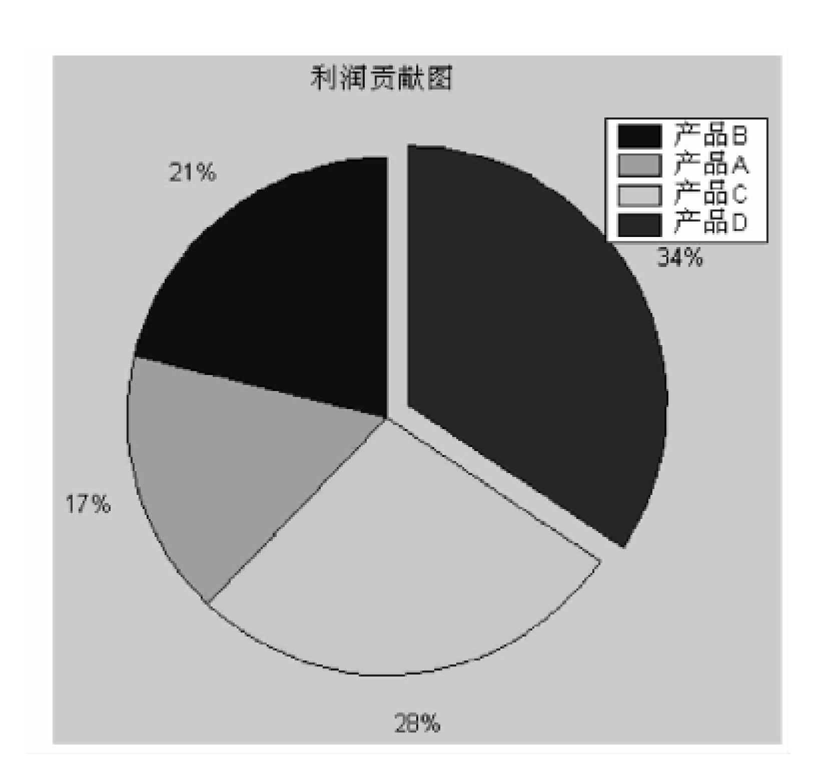


图 5-18　4种产品利润贡献图

【例 5-20】　某公司员工人数及每月工资支出见表5-10，试用饼图显示员工分类百分数及各类人员所占工资总额百分数。

表 5-10　某公司员工人数及每月工资支出

职　务	工　人	行政人员	高级主管	总经理
人数	150	50	12	3
月工资总额/元	150000	75000	48000	30000

解：

```
≫A = [150,50,12,3];                      %设置员工分类向量
≫B = [150000,75000,48000,30000];         %设置员工月工资总额向量
≫subplot(1,2,1);pie(A,[1,0,0,0])         % 分隔图形窗口成1行2列，当前为第1列；
                                            以向量A绘制饼图，向量[1,0,0,0]为分离
                                            向量，使工人这1块与饼图分开
≫subplot(1,2,2);pie(B,[1,0,0,0])         % 分隔图形窗口成1行2列，当前为第2列；
                                            以向量B绘制饼图，向量[1,0,0,0]为分离
                                            向量，使工人月工资总额这1块与饼图分
                                            开，其饼图如图5-19所示
```

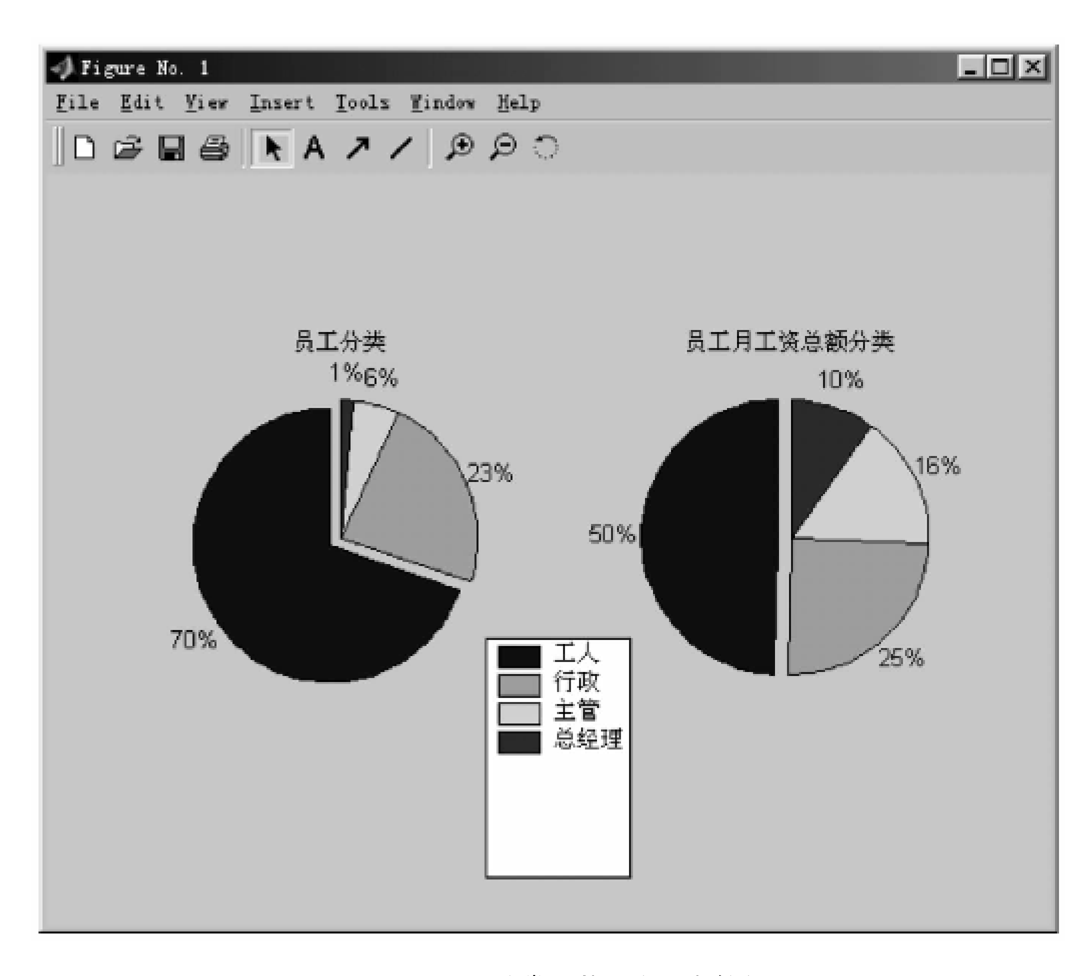


图 5-19 员工分类及其工资所占份额

5.7 阶梯图

阶梯图函数 stairs，对于绘制时间序列的样本数据是有用的。阶梯函数的书写格式为

```
stairs(Y)
stairs(X,Y)
stairs(…,LineSpec)
[xb,yb]=stairs(X,Y)
```

式中，X、Y 为同维向量，LineSpec 用来指定线型、标记和图形颜色。xb、yb 为返回向量，用来绘制线性阶梯图。

对于stairs(Y)用来绘制向量 Y 的阶梯图，横坐标则显示向量 Y 的序列。

对于stairs(X,Y)用来绘制向量 Y 的阶梯图，横坐标显示向量 X，向量 X 必须是单调递增的。

【例 5-21】 今有股市中某股票的采样数据如下：

```
Columns 1 through 10
    9.9100    9.6600   10.0200   10.0500    9.7700   10.2300   10.2300
  9.9900   10.0600   10.0300
Columns 11 through 20
    9.9600   10.1400    9.8800   10.4300    9.9700   10.0200   10.2100
```

```
  10.0100    9.9800    9.8300
   Columns 21 through 30
      10.0500    9.7300    10.1400    10.3200    9.8600    10.1700
10.2500    9.6800    9.7100    10.1100
   Columns 31 through 40
      9.9200    10.1300    10.1600    10.1400    10.2500    10.1300
10.2300    9.7500    9.9900    9.9600
   Columns 41 through 50
      9.6700    10.0500    9.7800    10.2800    9.8300    10.1000    10.0400
   9.8100    9.5600    9.9800
```

相应的采样时间为

```
   Columns 1 through 17
      2    4    6    7    9    10    12    15    17    20    21    22
25    28    30    32    35
   Columns 18 through 34
      36    38    39    40    41    42    44    47    51    55    58
60    62    63    66    67    71
   Columns 35 through 50
   74    75    78    79    81    85    88    88    92    94    95    96
   98    101    102    104
```

求此数据的阶梯图。

解：

```
>>pri = [9.91 9.66 10.2 10.05 9.77 10.23 10.23 9.99 10.06 10.03…
   9.96 10.14 9.68 10.43 9.97 10.0 10.21 10.1 9.98 9.83…
   10.05 9.73 10.14 10.32 9.86 10.17 10.25 9.68 9.71 10.11…
   9.92 10.13 10.16 10.14 10.25 10.13 10.23 9.75 9.99 9.96…
   9.67 10.05 9.78 10.28 9.83 10.1 10.04 9.81 9.56 9.98];     %输入股价向量
>>ti = [2 4 6 7 9 10 15 17 20 21 22 25 28 30 32 35 36 38 39 40 41 42 44 47 51 55 58 60 62 63 66…
   67 71 74 75 78 79 81 85 88 92 94 95 96 98 101 102 104];    %输入时间向量
>>h = stairs(ti,pri)                                           %绘制阶梯图，如图5-20
                                                                所示，并提取图形句
                                                                柄
h =
  104.0012
>>set(h,'linewidth',5,'color','r')                             %设置线宽为5及颜色为
                                                                红色
>>title'股价波动曲线'                                          %插入标题
>>xlabel('时间_分')                                            %插入x轴标签
>>ylabel('股价')                                               %插入y轴标签
```

```
>>grid on                                    %插入栅格线
```

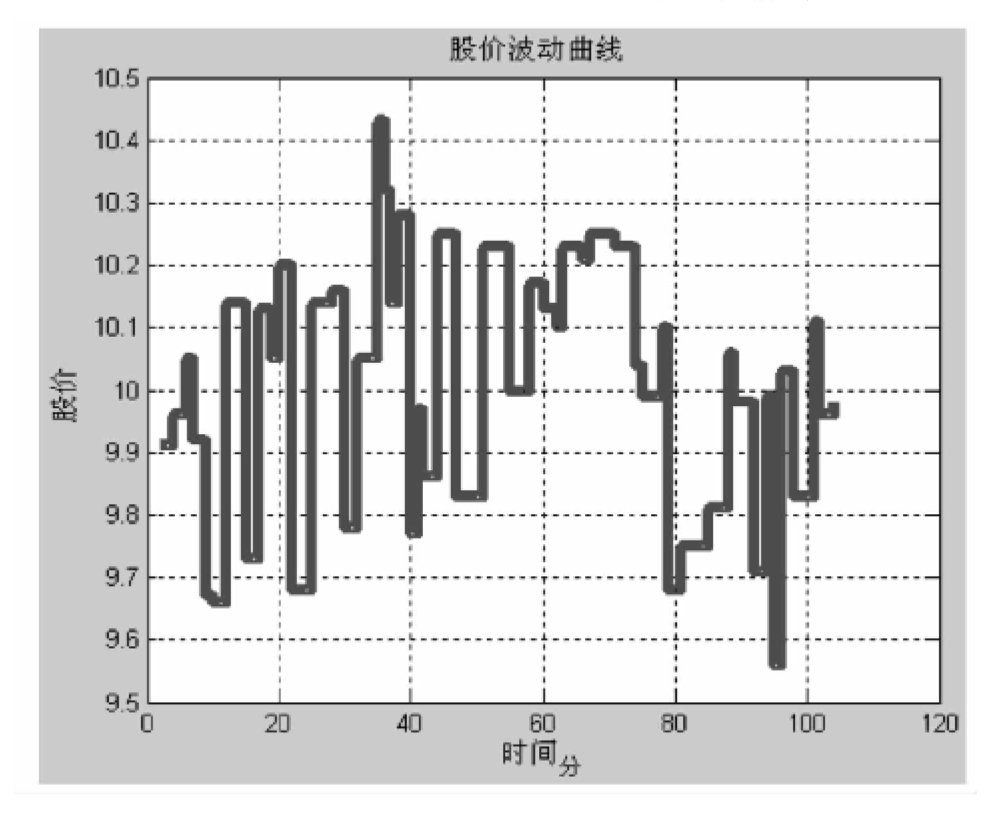


图 5-20 股价的阶梯图

【例 5-22】 某城市在职职工，月工资收入范围与所占总职工的人数百分比见表 5-11，试以阶梯图表示。

表 5-11 某市职工月工资收入与所占职工人数百分比

月收入/元	800	1000	1200	1500	2000	2500	3500	5000
占总职工人数（%）	10	13	21	33	12	8	2	1

解：

```
>>X = [800 1000 1200 1500 2000 2500 3500 5000];   %输入月收入向量
>>Y = [10 13 21 33 12 8 2 1];                      %输入百分比向量
>>h = stairs(X,Y)                                  % 绘制阶梯图，如图 5-21 所示，并
                                                     提取图形句柄
h =
  169.0004
>>set(h,'linewidth',8,'color','r')                 %设置图形线宽为 8，颜色为红色
>>grid on                                          %添加栅格线
>>title '收入分布'                                  %插入标题
>>xlabel('月收入')                                  %插入 x 轴标签
>>ylabel('百分比')                                  %插入 y 轴标签
```

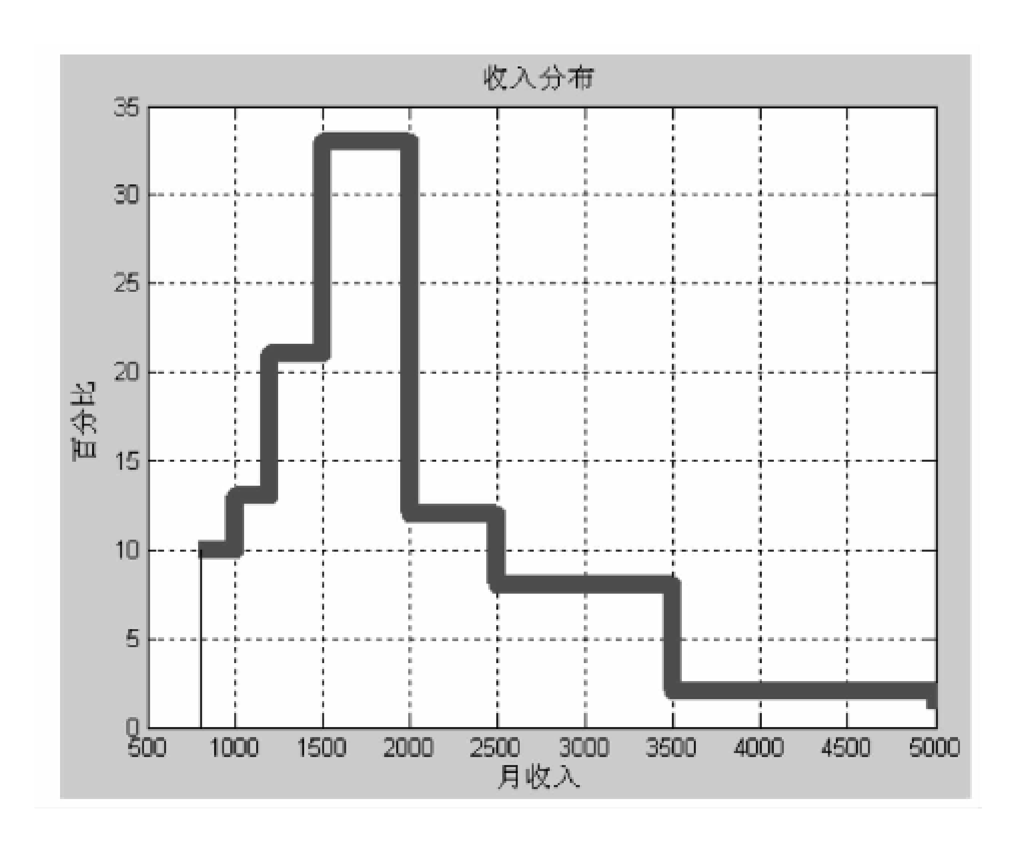


图 5-21 职工月收入统计图

5.8 茎干图

二维茎干图显示数据是用一根从 x 轴伸长的，并行于 y 轴的直线来表示。它的终端位置 y，用一个圆点或其他标记点来代表每个茎干的数值。茎干图的书写格式如下：

```
stem(Y)
stem(X,Y)
stem(…,'fill')
stem(…,LineSpec)
h = stem(…)
```

stem(Y)绘制以序列为横坐标，以向量 Y 的对应元素为纵坐标的茎干图。

stem(X,Y)中，当 X、Y 是向量或矩阵时，则分别用 X、Y 的列向量绘图，但 X、Y 的大小必须相等。作为补充，当 X 是向量而 Y 为矩阵时，则 Y 的行数应与length(X)相同。

stem(…,'fill')中，用以指定小圆点的充填颜色。stem(…,Line Spec)中，用以指定线型、标记点类型和颜色。

h = stem(…)则返回茎干图的图形对象句柄。

【例 5-23】 已知衰减正弦曲线y = exp(−0.5)xsin(x)，取向量$\boldsymbol{x}$ = [1:12]，绘制向量的茎干图。

解：

```
≫x = 1:12;                                 %输入向量 x
≫y = exp(-0.5* x).* sin(x);                %计算向量 y
≫stem(y)                                   %绘制茎干图
≫hold on                                   %图形保持
≫x1 = 0:0.1:12;                            %为绘制包络线，而设置细分向量 x1
≫y1 = exp(-0.5* x1).* sin(x1);             % 计算向量 x1 的对应值向量 y1
≫plot(x1,y1,':k')                          % 用虚线绘制线性图，如图 5-22 所示
```

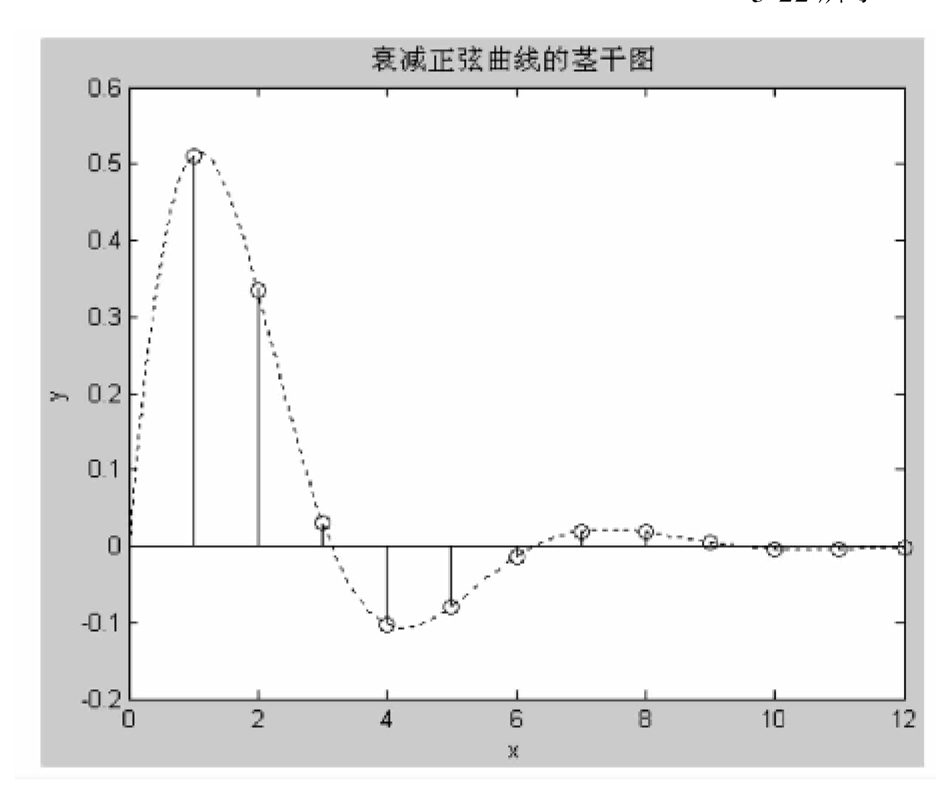


图 5-22　衰减正弦曲线的茎干图

【例 5-24】　某工程向银行贷款 2 亿，每年借贷 5 千万，5 年建成。建成后每年纯利润见表 5-12，试用茎干图表示。

表 5-12　某工程贷款及纯利润

年　　份	1	2	3	4	5	6	7	8	9	10	11	12
贷款（千万）	5	5	5	5								
建成后利润/（千万）					1.5	2.5	5	6	6.5	7	7	7

解：

```
≫A = [5,5,5,5,0,0,0,0,0,0,0,0];                 %设置贷款向量
≫B = [0,0,0,0,1.5,2.5,5,6,6.5,7,7,7];           %设置纯利润向量
```

```
≫subplot(2,1,1);axis([0 12 0 8]);hold on;stem(A,'filled')    % 绘图窗口设置，坐标设置，坐标保持，绘制A向量茎干图，圆点充填
≫subplot(2,1,2);stem(B,'filled');grid on                     % 绘图窗口设置，绘制B向量茎干图，圆点充填，加上栅格线，其图形如图5-23所示
```

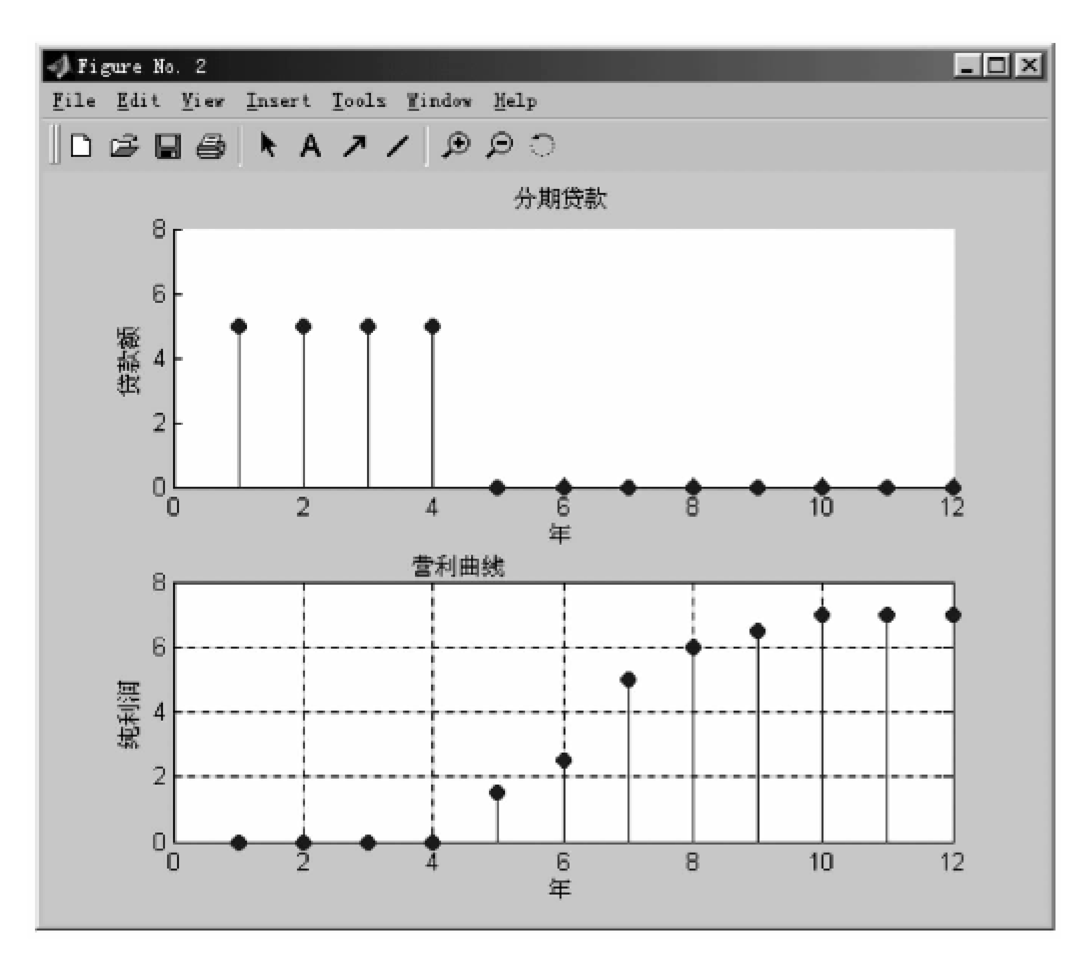


图 5-23 工程贷款及建成后逐年利润图

5.9 平面多边形的着色

为了丰富平面图形的色彩，MATLAB设置了着色函数fill，它能对多边形（包括封闭的曲线图形）进行着色。着色函数的书写格式如下：

```
fill(x,y,C)
fill(x,y,ColorSpec)
fill(x1,y1,C1,x2,y2,C2,…)
fill(…,'PropertyName',propertyValue,…)
h = fill(…)
```

式中，x、x1、x2为多边形顶点的横坐标向量；y、y1、y2为多边形顶点的纵坐标向量，C、C1、C2为色彩向量，它可以用色彩的简称表示，也可以用[R,G,B]表示。ColorSpec为色彩

规范，由表 5-13 表示。

表 5-13　色彩规范

R，G，B 值	简　　称	颜　色　名
1，1，0	y	黄色
1，0，1	m	紫红色、洋红色
0，1，1	c	青色、蓝绿色
1，0，0	r	红色
0，1，0	g	绿色
0，0，1	b	蓝色
1，1，1	w	白色
0，0，0	k	黑色

表达式h = fill(…)，则返回着色函数的句柄，用来修饰图形。

【例 5-25】 绘制正十六边形，在十六边形内涂以紫红色。

解：

```
≫t = pi/16:pi/8:2*pi;              %设置正十六边形的角度
≫x = cos(t);                       %设置正十六边形顶点的横坐标
≫y = sin (t);                      %设置正十六边形顶点的纵坐标
≫fill(x,y,'m'),axis square         %对正十六边形内涂以紫红色，坐标轴为正方形，如
                                    图 5-24 所示
```

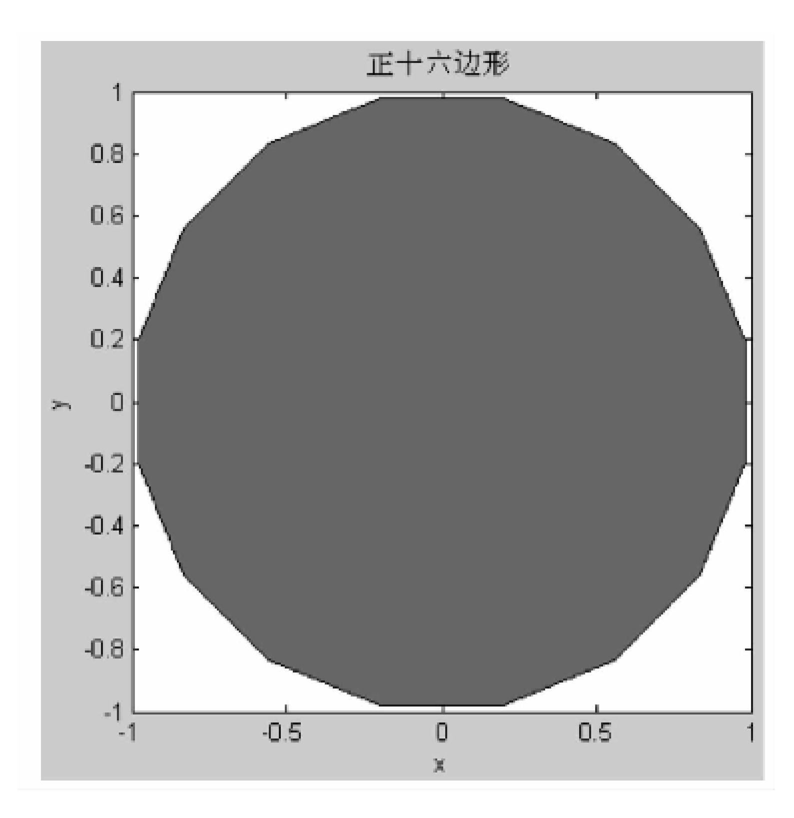


图 5-24　正十六边形的着色图

【例 5-26】 分别绘制正五边形着红色及五角星着绿色。

解：

```
>>theta = pi/2:2* pi/5:2* pi + pi/2       %设置五边形角度向量
theta =
    1.5708    2.8274    4.0841    5.3407    6.5973    7.8540
>>A = exp(i* theta);                      %五边形的顶点向量，在复平面上
>>x = real(A);                            %五边形顶点的实部向量
>>y = imag(A);                            %五边形顶点的虚部向量
>>subplot(1,2,1)                          %建立子图
>>fill(x,y,'r'),axis square               %对五边形着红色，设坐标轴正方形，如图 5-25 所示
>>title('正五边形')                        %设置标题
>>phi = pi/2:4* pi/5:4* pi;               %设置五角星的角度向量
>>B = exp(i* phi);                        %五角星的顶点向量
>>x1 = real(B);                           %五角星的顶点的实部向量
>>y1 = imag(B);                           %五角星的顶点的虚部向量
>>subplot(1,2,2)                          %设置子图
>>fill(x1,y1,'g')                         %绘制五角星，着绿色
>>axis square                             %坐标轴设为正方形
>>title('五角星')                          %设置标题
```

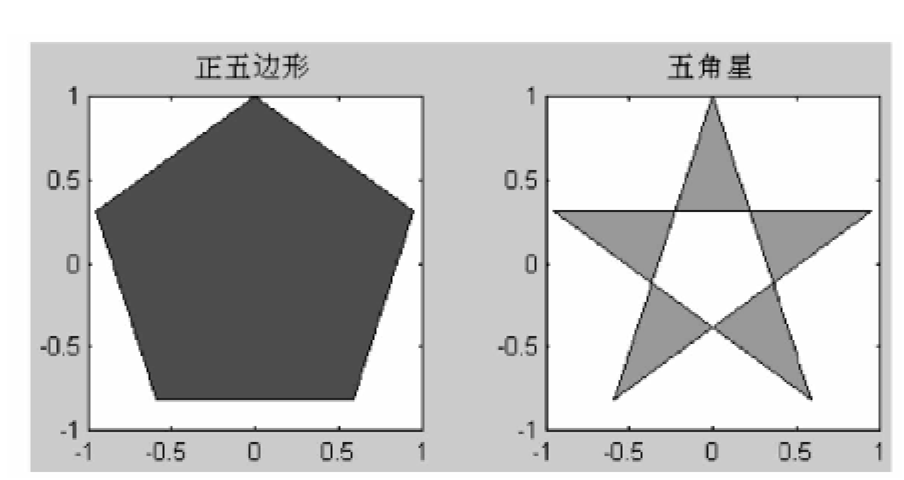


图 5-25　五边形与五角星的着色图

【例 5-27】　将抛物线 $y = 8x - x^2$ 与横坐标所围的面积涂绿色，与抛物线相切的圆 $(x - 10)^2 + (y - 10)^2 = r^2$ 涂红色。

解：

```
>>x = 0:0.2:8;                            %设置横坐标向量
>>y = 8* x - x.^2;                        %计算抛物线
>>plot(x,y)                               %绘制抛物线，如图 5-26 所示
>>axis([0,16,0,16])                       %设置坐标轴
>>x = 10 - 3.84:0.02:10 + 3.48;           %设置画圆的 x 轴向量，3.84 为相切圆半径 r
>>y1 = 10 + sqrt(3.48^2 - (x - 10).^2);   %计算上半圆
```

```
>>y2 = 10 - sqrt(3.48^2 - (x - 10).^2);                              %计算下半圆
>>hold on                                                            %保持原有图形
>>plot(x,y1,x,y2)                                                    %绘制圆
>>fill(x,y1,'r',x,y2,'r')                                            %圆着色
Warning: Imaginary parts of complex X and/or Y arguments ignored. %警告：x、y 的虚部忽略
>>x = 0:0.1:8;                                                       %重设横坐标向量
>>y = 8* x - x.^2;
>>fill(x,y,'g')                                                      %对抛物线部分涂绿色
>>axis square                                                        %设置坐标轴正方形
>>title('抛物线与圆相切的着色图')                                    %设置标题
>>xlabel('x')                                                        %设置坐标轴标签
>>ylabel('y')
```

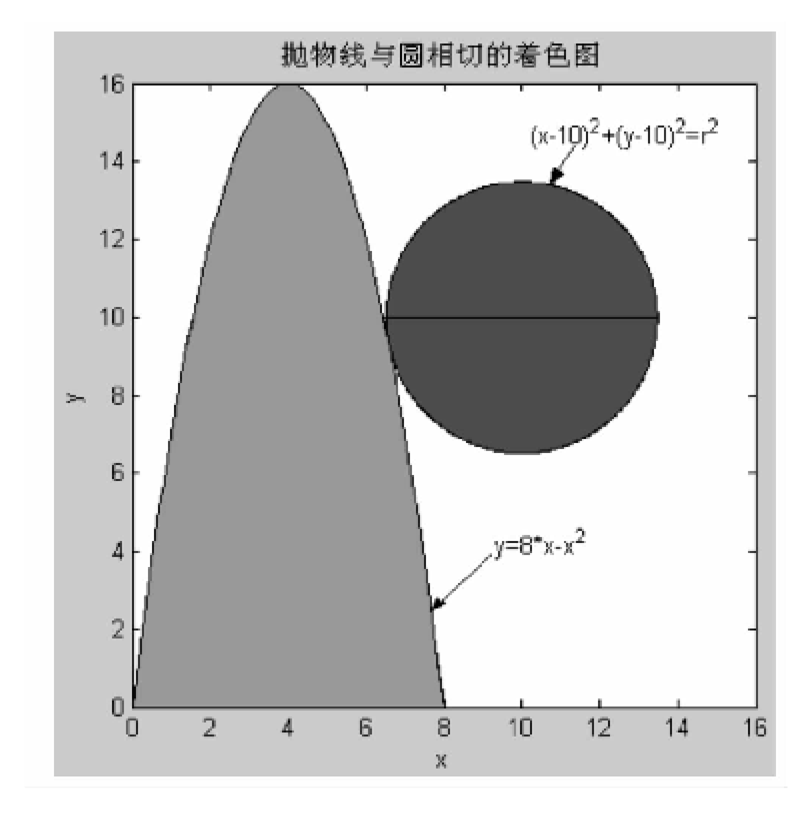


图 5-26　抛物线与相切圆的着色图

5.10　三维特殊图形

三维立体图的绘制，其中一部分绘图函数可在二维平面图形绘图函数的基础上添加数字 3 来实现。例如三维线性图函数为 plot3，三维条形图为 bar3、barh3，三维饼图为 pie3，三维散点图为 scatter3，三维茎干图为 stem3 等。但是三维立体图又增加了不少新的功能。如网格图，表面图，视角控制，光照，透明度控制等。

5.10.1　三维线性图

三维线性图的书写格式为

$$plot3(X,Y,Z)$$

式中，X、Y、Z 为同维的向量或矩阵。有关线的规范的设置与二维的线性图完全相同。这里不再重复。

【例 5-28】 已知空间螺旋线方程式为

$x=0.1\exp(t/20)\cos(2t)$，$y=0.1\exp(t/20)\sin(2t)$，

$z=t$，设置t = 0: pi/50: 10* pi，求三维线性图。

解：

```
>>t = 0: pi/50: 10* pi;                    %设自变量向量
>>x = 0.1* exp(t/20).* cos(2* t);          %计算 x 向量
>>y = 0.1* exp(t/20).* sin(2* t);          %计算 y 向量
>>plot3(x,y,t),grid on                     %图形如图 5-27 所示
```

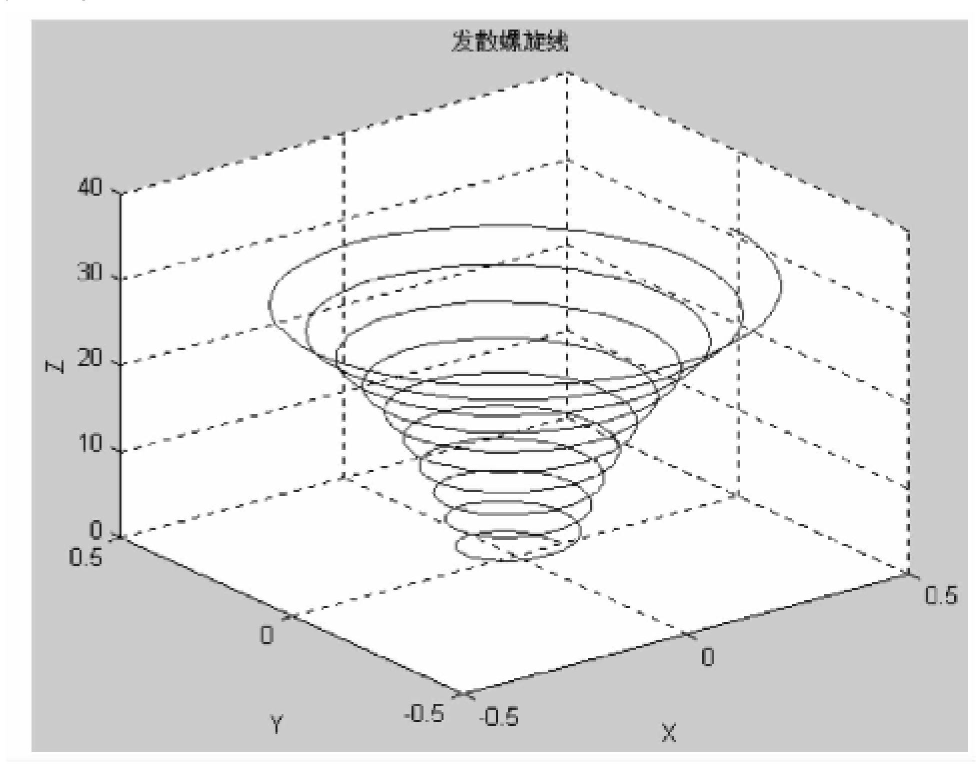


图 5-27 发散的螺旋线

5.10.2 三维条形图

三维条形图的书写格式为

$$bar3(Y)$$

$$bar3(X,Y)$$

$$bar3(\cdots,width)$$

bar3(Y)绘制向量 Y 每一个元素的条形图，立体条形的幅值代表元素值，条形图的下标为向量 Y 的序列。假若 Y 为矩阵，则产生代表矩阵每一行元素值的条形组，条形组的横坐标是显示矩阵的行数。

对于bar3(X,Y)，则绘制向量 Y 的每一个元素在指定 X 位置的条形图，向量 X 的值必须是单调递增的。假若 Y 为矩阵，则一串立体条形代表矩阵 Y 的行元素，放置于横坐标 X

的位置。

对于bar3(…,width)用来设置立体条形的宽度，默认时为0.8，当宽度设为1时，则条形一个紧挨着一个，没有间隙。若没有指定宽度，则条形组之间有细小的分离。

【例5-29】 根据1990~2001年国民经济和社会发展统计报告，历年来工农业生产总值和消费品价格上涨幅度见表5-14，请用三维直方图表示。

表5-14　1990~2001年国民经济和社会发展统计报告

年　份	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001
总产值（千亿）	17.4	19.58	23.94	31.38	43.8	57.73	67.79	74.77	79.55	82.05	89.4	95.93
消费品上涨百分数（%）	3.1	3.4	10.7	19.6	24.8	16.5	8.3	2.8	-0.8	-1.4	0.4	0.7

解：

```
>>year = [1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001];
                                                %年份向量
>>GDP = [17.4 19.58 23.94 31.38 43.8 57.73 67.79 74.77 79.55 82.05 89.4 95.93];
                                                %生产总值向量
>>consu = [3.1 3.4 10.7 19.6 24.8 16.5 8.3 2.8 -0.8 -1.4 0.4 0.7];
                                                %消费品价格上涨幅度向量
>>colormap(spring)                              %设置色彩
>>bar3(year,GDP,0.3)                            %绘制三维直方图如图5-28所示
```

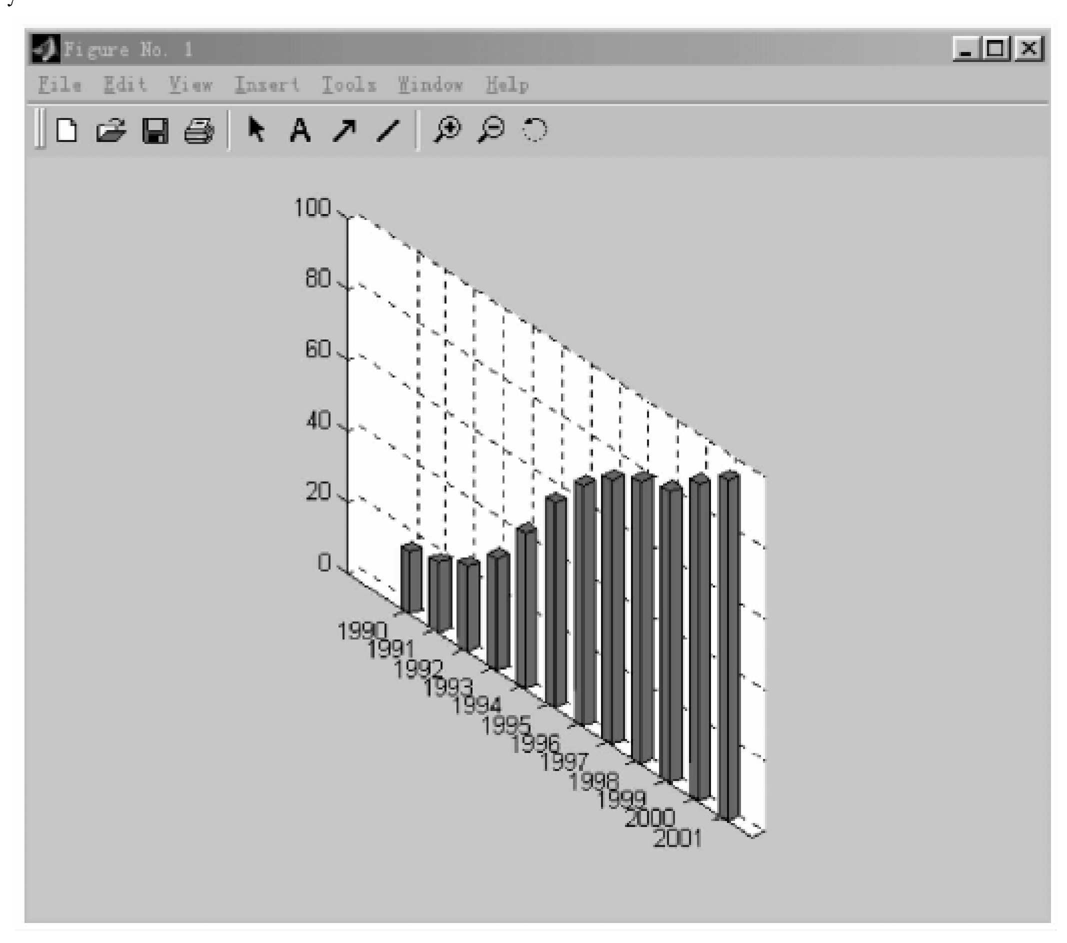


图5-28　1990~2001年工农业生产总值的三维直方图

```
≫figure                                 %创建另一图形窗口
≫colormap([0 1 1])                      %设置色彩
≫bar3(year,consu,0.3)                   %绘制三维直方图如图 5-29 所示
```

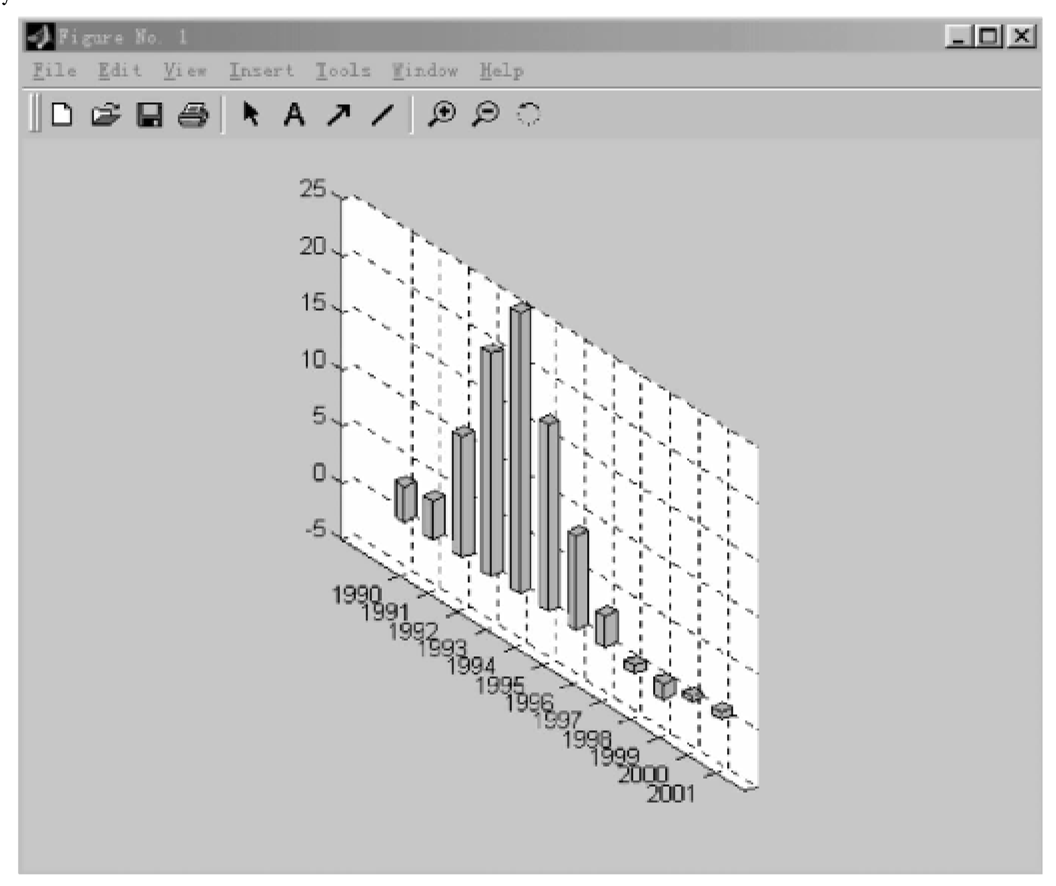


图 5-29　1990～2001 年消费品价格上涨幅度的三维直方图

5.10.3　三维散点图

三维散点图的书写格式为

scatter3(x,y,z)

scatter3(x,y,z,s,c)

式中，x、y、z 为三维坐标向量，它们必须有相同的长度。s 是指圆圈标记点的面积，面积定为点宽的平方，它可以是标量，若是向量，则必须与 x、y、z 大小相同，c 用来确定标记的颜色。若是向量，则必须与 x、y、z 具有相同的大小。在标记类型默认的情况下，scatter3 绘出的是圆圈图。

【例 5-30】　利用三维散点图，绘制模拟的喷泉。

解：为模拟喷泉，在 M 文件编辑器编制程序如下，程序名为 fountain_prog.m，图形名为 fountain.fig

```
clear                                   %清内存
for n=0:0.5:15                          %设置喷泉层次
  r=5+5*n;                              %设置喷泉的内、外径
    theta=0:pi/12:2*pi;                 %设置喷嘴数，计 25 个
```

```
    x = r* cos(theta);                          %计算散点 x 轴坐标
    y = r* sin(theta);                          %计算散点 y 轴坐标
    z = 50* ones(size(x))* n - 2* n.^2;         %计算散点 z 轴坐标，近似抛物线
    c(1) = 0; c(2) = 0; c(1) = 0;               %设置颜色，黑色
    scatter3(x, y, z, 3, c, 'filled')           %绘制散点图，如图 5-30
    hold on                                     %图形保持
end                                             %循环
axis([-80, 80, -80, 80, 0, 320])                %设置坐标轴范围
```

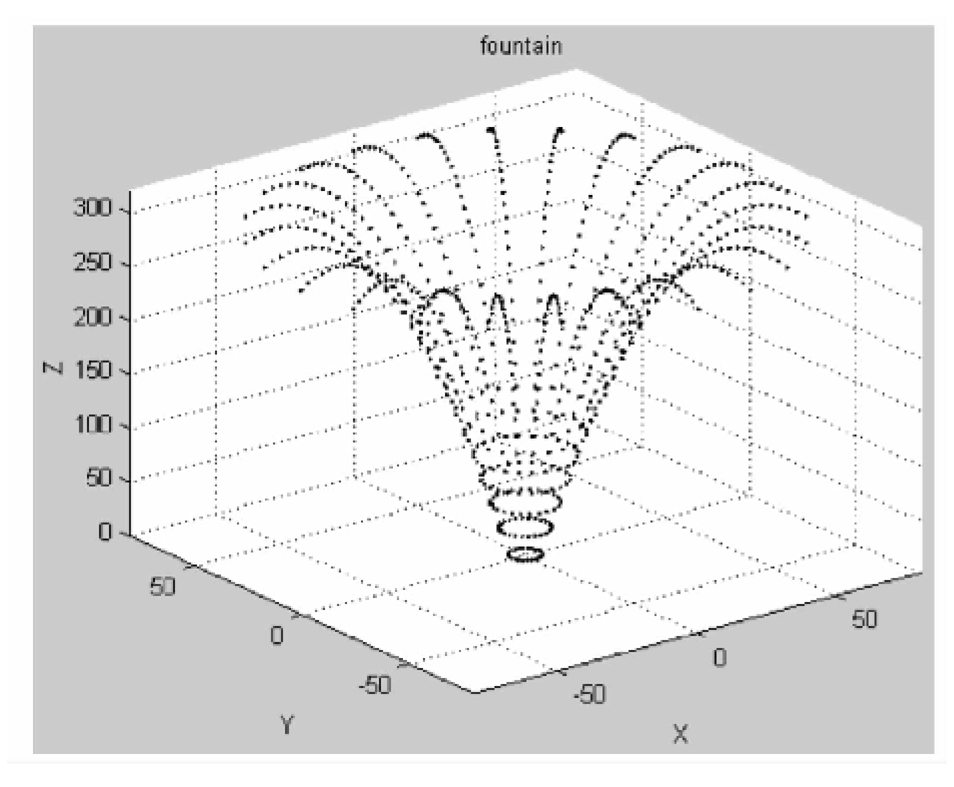


图 5-30　喷泉模拟图

5.11　三维网格图

网格图是把观察范围的 x、y 平面设置成均匀分布的网格，在网格的交叉点取 z 轴的点，$z = f(x, y)$，随后把相邻的空间点用直线联起来，则构成单元网格，单元网格可以是三角形的或四边形的，观察范围内的单元网格的总和构成三维网格图。单元网格为四边形的网格图的书写格式为

```
mesh(X, Y, Z, C)
meshc(X, Y, Z, C)
meshz(X, Y, Z, C)
h = mesh(…)
h = meshc(…)
h = meshz(…)
```

式中，X、Y 为坐标轴取值向量或与 Z 同维的矩阵，Z 为 X、Y 平面上的函数值矩阵，C 为色彩向量，当 C 默认时，网格图的色彩随 Z 的高度而变。

meshc 是在网格图的基础上添加等高线，等高线是画在网格图下的 X、Y 平面上。

meshz 是在网格图下添加门帘线。

式中，h 用来获取图形句柄，用于图像的修饰。对于三角形网格图，请读者参阅 help trimesh。

【例 5-31】 已知三维方程式 $z = \exp(-x^2 - y^2)\sin^2 x$，求三维网格图。

解：

```
≫[X,Y]=meshgrid(-3:0.2:3);                          %设置矩阵网格
≫Z=exp(-X.^2-Y.^2).*sin(X).^2;                      %计算矩阵 Z
≫mesh(X,Y,Z)                                         %绘制网格图，如图 5-31 所示
≫title('Z=exp(-X.^2-Y.^2).*sin(X).^2，的网格图');   %设置图形标题
≫xlabel('X');                                        %设置坐标标签
≫ylabel('Y');
≫zlabel('Z');
≫axis([-3,3,-3,3,0,0.3])                             %设置坐标轴范围
≫max(max(Z))                                         %计算最大值
ans =
    0.2713
```

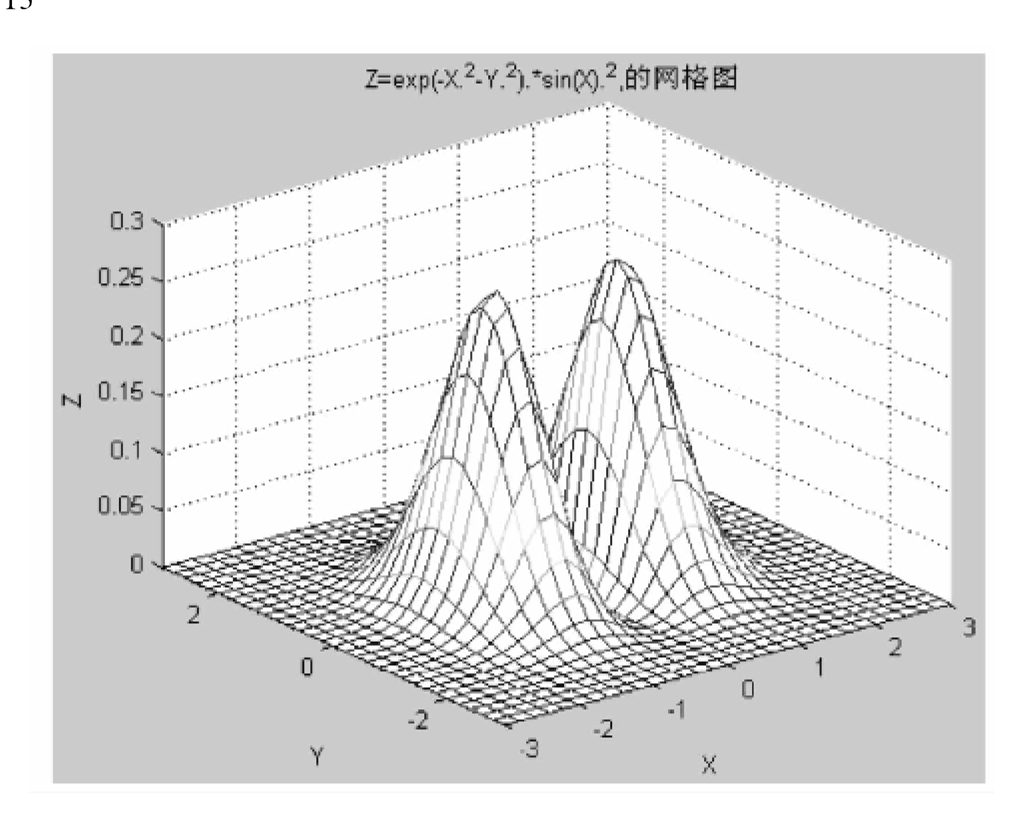


图 5-31 $z = \exp(-x^2 - y^2)\sin^2 x$ 的网格图

【例 5-32】 已知三维方程式 $z = \exp(-x^2 - y^2)\sin x$，求三维网格图，并带有等高轮廓线。

解：

```
≫[X,Y]=meshgrid(-3:0.2:3);                          %设置矩阵网格
```

```
≫Z = exp(－X.^2－Y.^2).* sin(X);                    %计算矩阵 Z
≫meshc(X,Y,Z)                                      %绘制带有等高线的网格图，如
                                                     图 5-32 所示
≫axis([－3,3,－3,3,－0.3,0.3])                      %设置坐标轴范围
≫title('Z = exp(－X.^2－Y.^2).* sin(X)，的网格图');  %设置图形标题
≫xlabel('X');                                      %设置坐标轴标签
≫ylabel('Y');
≫zlabel ('Z');
≫max (max (Z))                                     %计算最大值
ans =
    0.3939
```

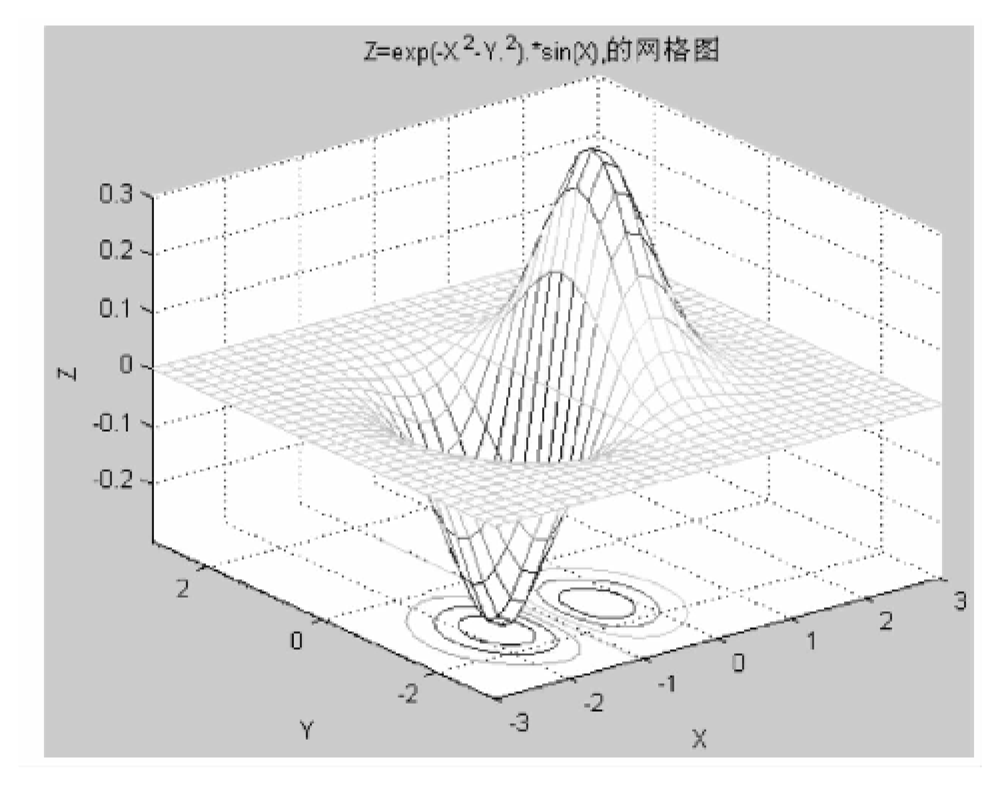


图 5-32　$z = \exp(-x^2 - y^2)\sin x$ 带有等高线的网格图

5.12　三维表面图

上一节讲述的三维网格图，它的网格线是彩色的，并且它的色彩随着 z 轴的高度而改变。本节叙述的三维表面图，它的网格线和网格单元表面都随着 z 轴坐标的高度而改变，因此它的立体感将更强。三维表面图的书写格式为

```
surf(Z)
surf(X,Y,Z)
surf(X,Y,Z,C)
surfc(…)
h = surf(…)
```

h = surfc（…）

式中，Z 为矩阵，X，Y 为坐标向量或与 Z 同维的矩阵。C 为色彩向量，色彩向量的行数为（$m-1$）（$n-1$），它对应网格的单元数，其中 m 为 Y 向量的长度，n 为 X 向量的长度，[m，n] = size（Z）。色彩向量的列数为 3，由 R、G、B 来表示。当色彩向量默认时，则 colormap（色彩图表自动设置）。Colormap 亦可由 colormapeditor（色彩图表编辑器）来设置。

surf（Z）用来绘制矩阵 Z 的表面图，矩阵的列数作为 x 轴坐标，矩阵的行数作为 y 轴坐标。Z 的元素是单值的，它指定了色彩数据，并且它的幅值正比于它的颜色。

surf（X，Y，Z）产生由向量 X，Y 几何网格下的表面图。[m，n] = size（Z），而 m = length（Y），n = length（X）.。

surf（X，Y，Z，C）建立的表面图，它的色彩由矩阵 C 定义。MATLAB 从当前的色图（colormap）执行线性转换。

surfc（……）绘制带有等高线的表面图。这等高线是绘制在表面图下的 x，y 平面上。

h 为图形的句柄。用来修饰图形用。

【例 5-33】 已知矩阵 A 为 9 阶 pascal 矩阵，求表面图。

解：

```
>>A = pascal（9）；                %9 阶 pascal 矩阵
>>surf（A）                        %绘制表面图，如图 5-33 所示
>>title（'pascal（9）'）；          %设置标题
>>xlabel（'X'）；                   %设置坐标轴标签
>>ylable（'Y'）；
>>zlabel（'Z'）；
```

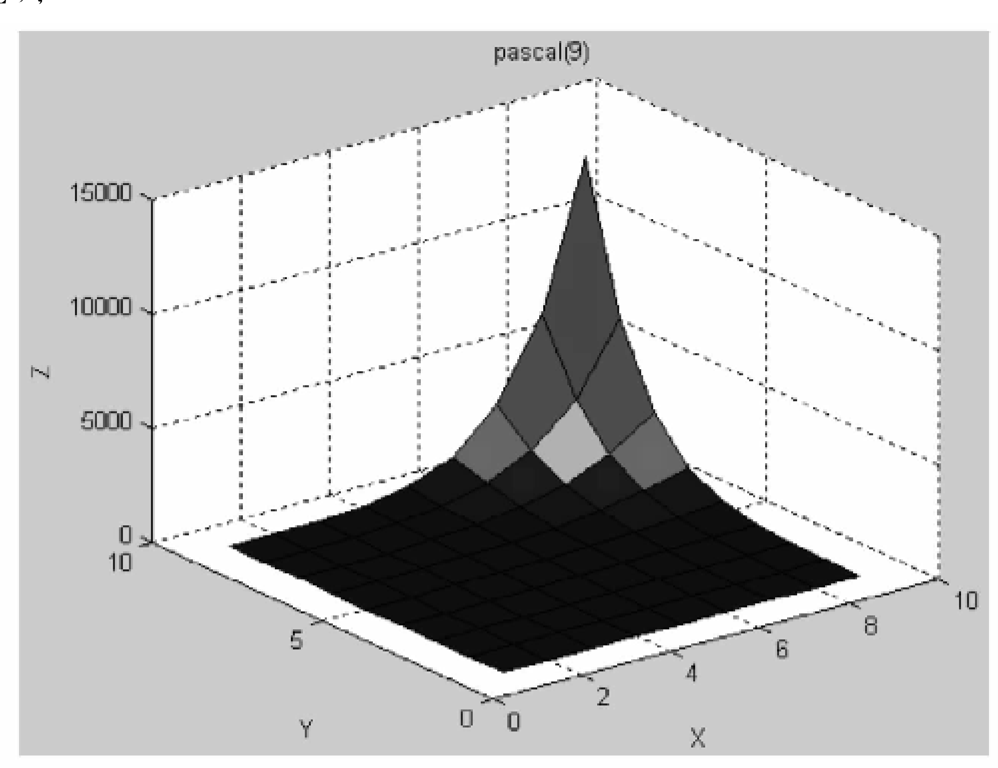


图 5-33　9 阶 pascal 矩阵的表面图

【例 5-34】 已知 ***B*** 为 9 阶魔方阵，求矩阵 ***B*** 的表面图。

解：

```
>>B = magic(9);                              %建立魔方矩阵
>>surf(B)                                    %绘制表面图，如图 5-34 所示
>>title('magic(9)的表面图')                  %设置标题
>>xlabel('X')                                %设置坐标标签
>>ylabel ('Y')
>>zlabel ('Z')
```

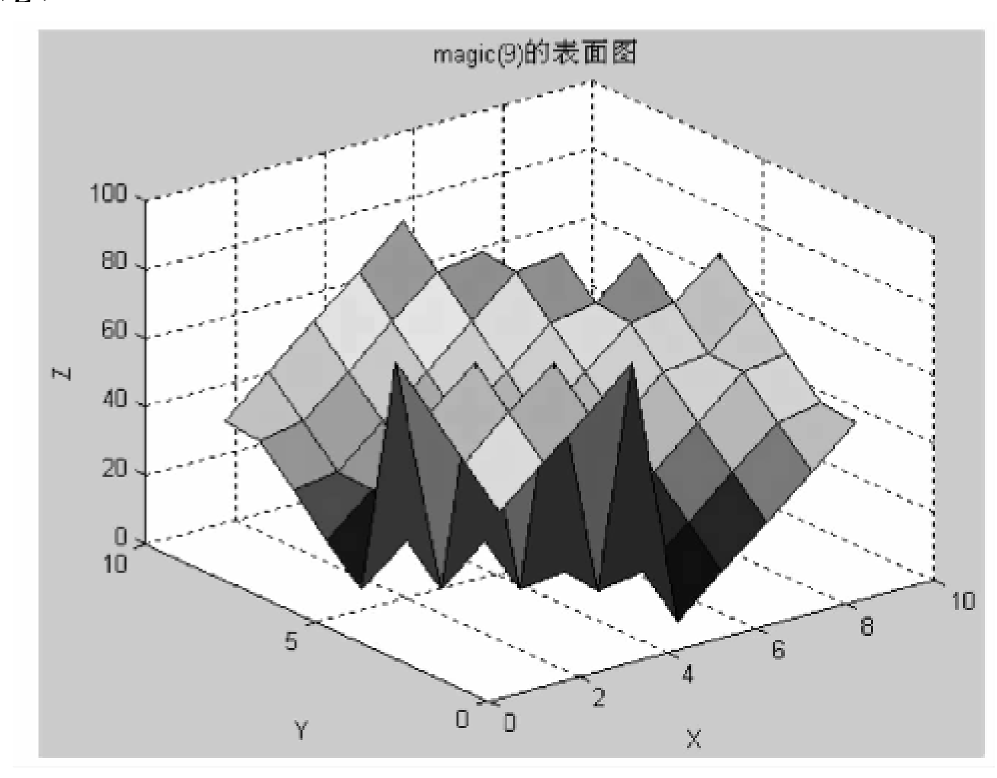


图 5-34　9 阶魔方阵的表面图

【例 5-35】 已知三维函数 $Z=\exp[-(0.15X)^2-(0.15Y)^2]\sin X\sin y$，求表面图并带有等高线。

解：

```
>>[X,Y]=meshgrid(0:0.25:10);                 %设置网格矩阵
>>Z=exp(-(0.15*X).^2-(0.15*Y).^2).*sin(X).*sin(Y);    %计算矩阵 Z
>>surfc(X,Y,Z)                               %绘制带有等高线的网格图，如图 5-35 所示
>>title(' Z=exp(-(0.15*X).^2-(0.15*Y).^2).*sin(X).*sin(Y)的表面图')
                                             %设置标题
>>xlabel('X')                                %设置坐标标签
>>ylabel('Y')
>>zlabel('Z')
```

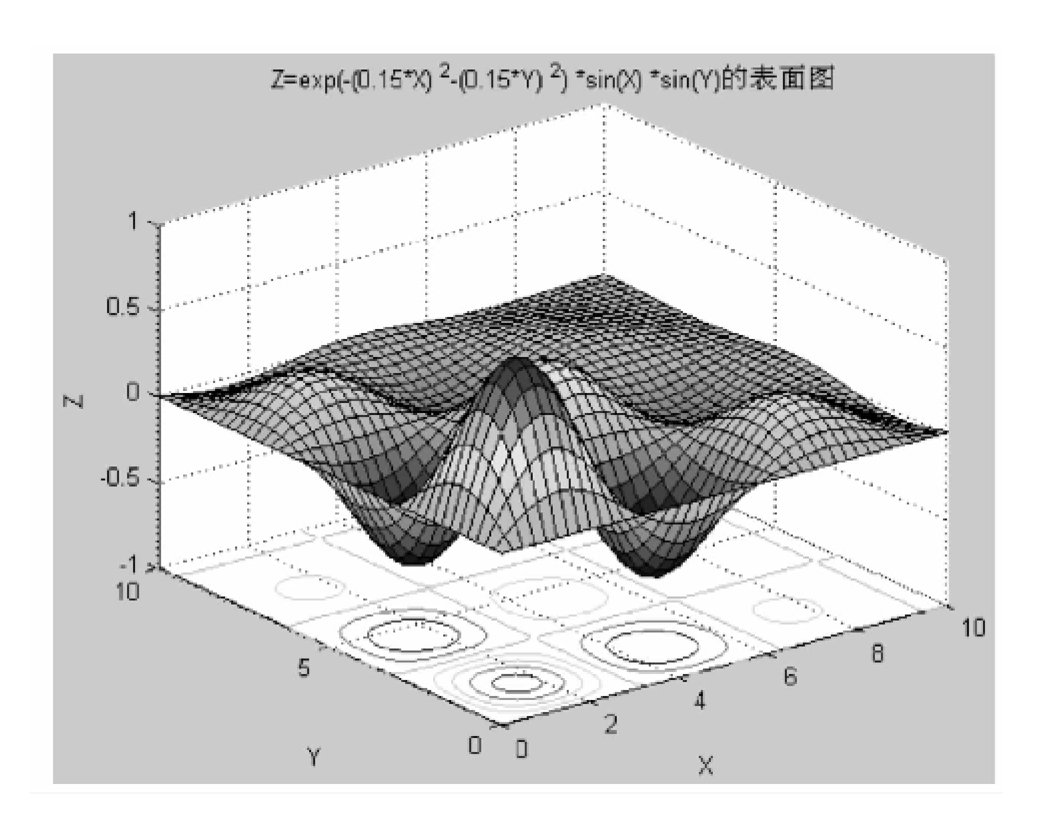


图 5-35　带有等高线的表面图

5.13　简易表面图

为了绘制表面图的方便，而设置了简易表面图函数 ezsurf，它不用设置自变量的间隔向量，线宽、标记点、颜色，只要知道函数的符号表达式即可绘出函数图形。ezsurf 的书写格式为

```
ezsurf(f)
ezpsurf(f,domain)
ezpsurf(x,y,z)
ezpsurf(x,y,z,[min max])
```

式中，f 为符号函数表达式，domain 为自变量变化区间。在默认的情况下变量 x、y 的变化区间均为[$-2\pi,2\pi$]。

eurf(x,y,z)绘制含有 2 个参变量 s、t 的表面图。其中x = x(s,t)， y = y(s,t)， z = z(s,t)，s 和 t 均在区间[$-2\pi,2\pi$]。

参变量 s、t 亦可自行设置变化区间为[smin,smax,tmin,tmax]。

对于变量 x、y 均为参变量 t 的函数，则可在书写格式为[tmin,tmax]。

【例 5-36】　已知方程式为 $z = x\exp(-x^2 - y^2)$，求该函数的三维表面图，坐标范围为[$-3,3$]。

解：

```
>>ezsurf('x* exp(-x^2-y^2)',[-3,3],[-3,3]);grid on    % ezsurf 为绘制函数曲面图命令，引
                                                       号内为函数表达式，方括号内为自
                                                       变量取值区间，函数图形如图 5-36
```

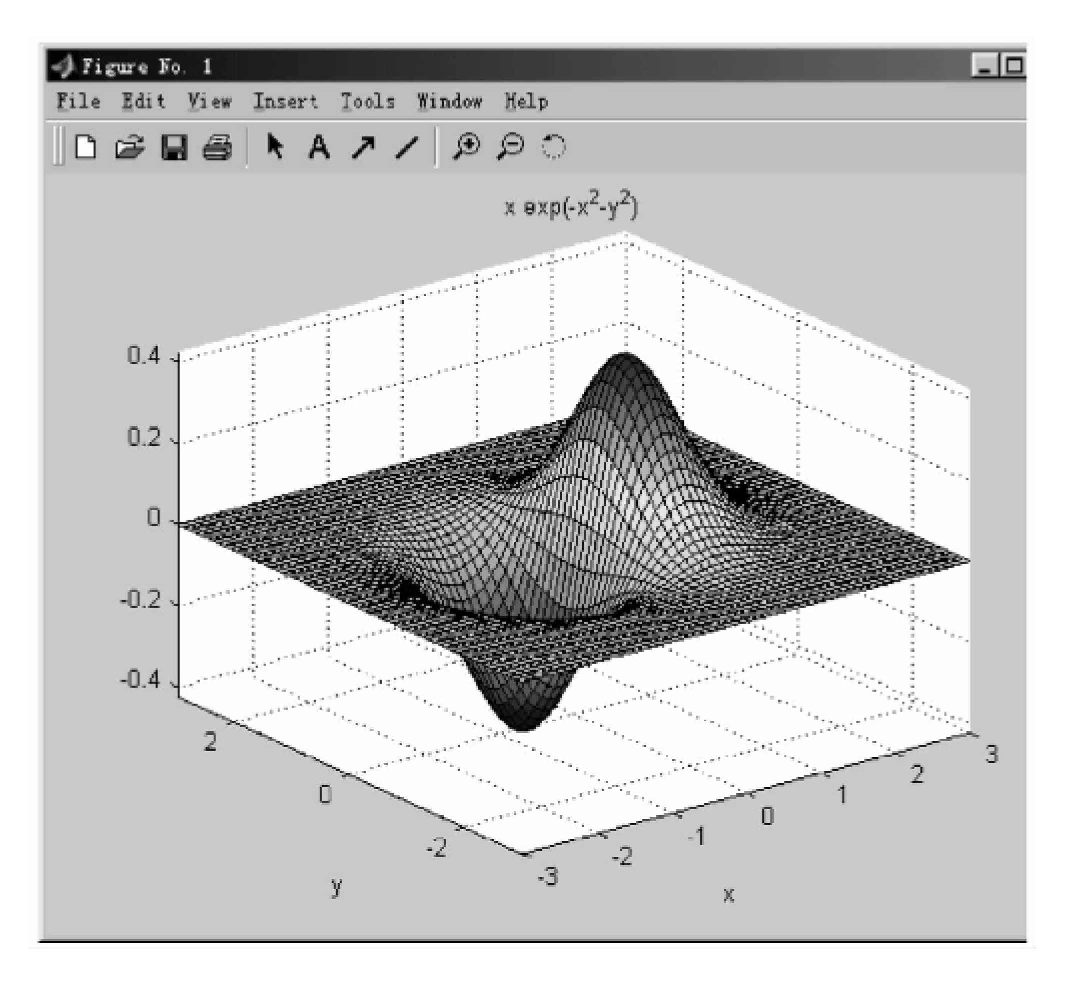


图 5-36 方程式 $z = x\exp(-x^2-y^2)$ 的三维曲面图

【例 5-37】 已知三维函数 $x = r\cos(\text{theta})$，$y = r\sin(\text{theta})$，$z = r^2$，求该函数的简易表面图。

解：

```
>>syms r theta                    %设置符号变量
>>x = r* cos(theta);              %输入变量 x 的符号表达式
>>y = r* sin(theta);              %输入变量 y 的符号表达式
>>z = r^2;                        %输入变量 z 的符号表达式
>>ezsurf(x,y,z)                   %绘制简易表面图，如图 5-37 所示
```

5.14 柱形立体图

柱形立体图是由柱形图函数 cylinder，产生 ***X***、***Y*** 和 ***Z*** 的坐标矩阵，随后由 mesh 或 surf 产生柱形网格图或表面图。柱形图函数 cylinderr 的书写格式为

```
[x,y,z] = cylinder(r)
[x,y,z] = cylinder(r,n)
cylinder(…)
```

式中，r 为圆柱体半径向量，在默认的情况下圆柱体为 20 等分。n 为指定等分数。x、y、z 为代表柱形体的坐标矩阵。

cylinder(r)返回由半径 r 定义的圆柱体的轮廓曲线的三维坐标。

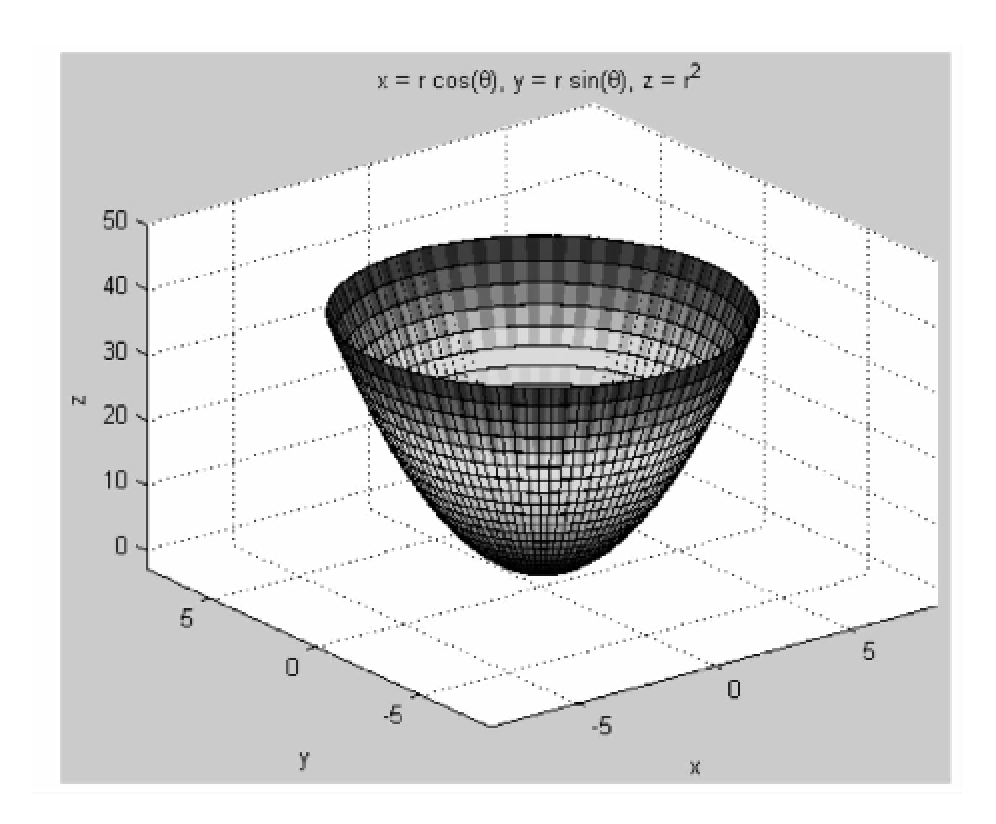


图 5-37 简易表面图（抛物面）

cylinder(r,n)则返回由半径 r 定义的圆柱体半径，将圆柱体等分成 n 的三维坐标。

cylinder(…)则无输出变量，并且直接由 surf 表面图绘出柱形图。

【例 5-38】 用三维柱形图，绘制灯笼。

解：

```
≫zeta = - pi/2:pi/12:pi/2;          %设置角度向量
≫r = 0.5 + cos(zeta);               %设置圆柱体半径向量
≫r = [0.5,r,0.5];                   %延伸灯笼开口段
≫[x,y,z] = cylinder(r,12);          %计算圆柱体三维坐标。圆柱体 12 等分
≫surf(x,y,z),grid on                %绘制三维表面图。如图 5-38 所示
```

【例 5-39】 用三维柱形图，绘制漏斗。

解：

```
≫theta = - pi/2:pi/12:0;            %设置角度向量
≫v = 1 + 3* cos(theta);             %设置漏斗体向量
≫v1 = 0.5:0.1:1;                    %设置漏斗嘴向量
≫V = [v1,v];                        %漏斗体合成向量
≫[x,y,z] = cylinder(V,120);         %计算柱形图的三维坐标
≫surf(x,y,z),grid on                %绘制表面图，如图 5-39 所示
≫title('漏斗')                      %设置标题
≫xlabel('X')                        %插入坐标轴标签
≫ylabel('Y')
≫zlabel('Z')
```

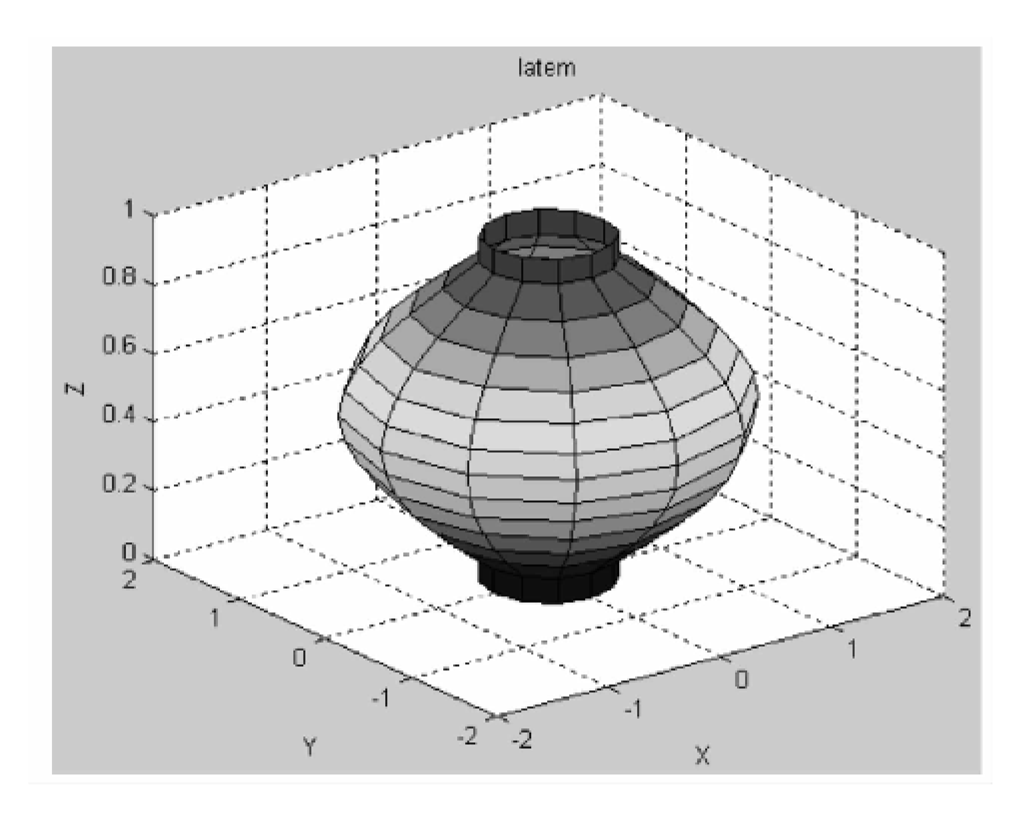


图 5-38 灯笼的表面图

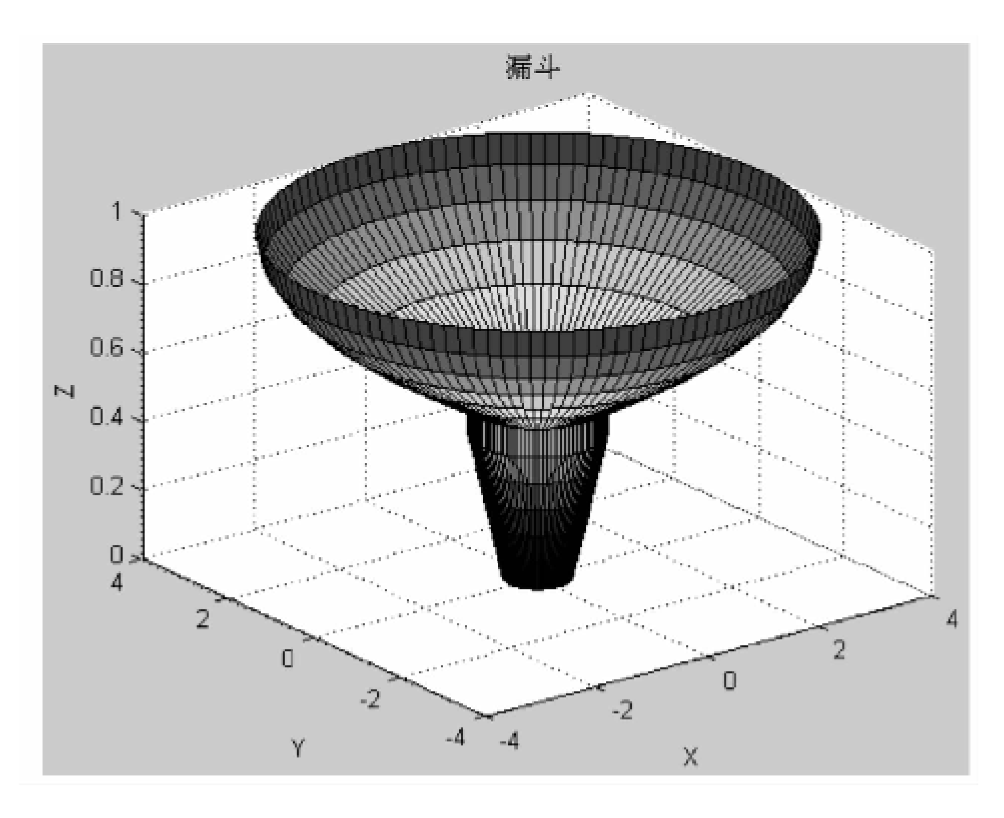


图 5-39 漏斗表面图

5.15 图形格式的设置

当使用 MATLAB 绘图函数，绘制好图形以后，为了丰富图形的内涵，还必须对图形添加标题、坐标轴标签、文字说明、图例、辅助线、指示线等，如图 5-40 所示。

MATLAB 可以通过三种方式对图形进行标注。

(1) 从图形中直接标注。点击图形窗口中的插入菜单 (Insert),在插入菜单中有子菜单 *x*、*y*、*z*、label (坐标轴标签),title (标题),legend (图例),colorbar (色彩分层),arrow (箭头),line (辅助线),text (文本说明),axes (坐标轴设置),light (光照设置) 等供标注或设置。

(2) 使用图形标注函数进行标注。使用图形标注函数,可以直接从程序中编写,当执行程序后,图形中自动添加了图形标注。常用图形标注函数见表 5-15:

表 5-15　常用图形标注函数

函数名	说明
title	建立图形标题
xlabel	建立 *x* 轴标签
ylabel	建立 *y* 轴标签
zlabel	建立 *z* 轴标签
legend	在图形中添加图例
text	在指定位置添加文本说明
gtext	使用鼠标在图形某个位置插入文本
grid	栅格线显示控制,grid on 为显示栅格线,grid off 为取消栅格线
hold	图形保持控制。hold on 保持当前图形,hold off 为取消图形保持
subplot	在图形窗口,建立子图窗口
figure	新建图形窗口
plotedit	打开图形编辑窗口
axes	建立坐标轴图形对象
axis	建立坐标刻度和范围

(3) 使用图形的属性编辑器。在当前图形的菜单栏中选择 edit/figure prorerties,并点击,即进行图形的属性编辑器。在编辑器里能对图形、线条、坐标、颜色、视角、光照进行编辑和设置。

【例 5-40】 已知某公司生产 A、B、C、D、E、F 共 6 种产品,年利润分别为 45、75、105、175、85、90 万元,用饼图显示产品的贡献,并为图形添加标题和图例。

解:

```
>>V = [45,75,105,175,85,91];
>>V1 = V/sum(V)
V1 =
    0.0781    0.1302    0.1823    0.3038    0.1476    0.1580
>>[m,i] = max(V1);
>>explode = zeros(size(V1));
>>explode(i) = 1;
>>pie(V1,explode)
>>title('产品利润贡献图')
```

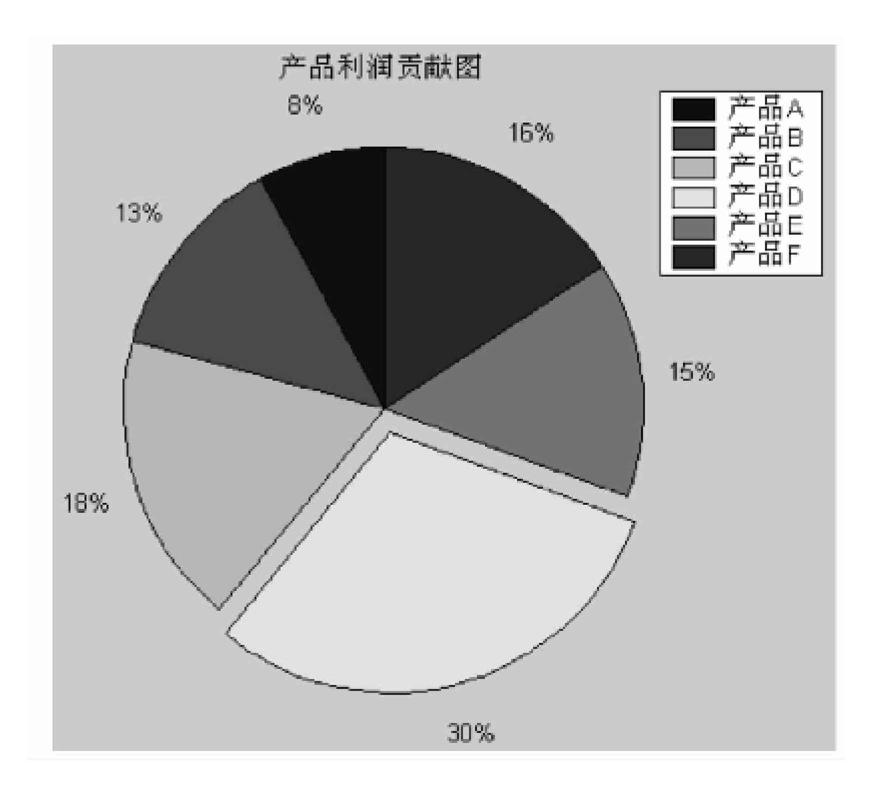


图 5-40　图形标题和图例的设置

≫legend('产品 A','产品 B','产品 C','产品 D','产品 E','产品 F')

5.16　视角与色彩控制

MATLAB 能够控制观察在坐标轴上立体图的方位。用户可以指定观察点，观察目标和观察的方位。这些观察特性也可以通过设置图形属性来达到。你可以指定这些图形的特性值通过图形属性编辑器，或者使用 view 命令去选择观察方向以及依靠 MATLAB 属性选择功能定义一个合适的视图。

view 命令是一个视角向量。它的一个元素指定了方位角 az，它是视角向量在 xy 平面上投影线与负 y 轴的夹角，逆时针方向为正。另一个元素是 el，它指定了仰角，这仰角是视角向量与其在 x，y 平面上投影线的夹角。

view 的书写格式为

view(az,el)——设置方位角和仰角；

view([az,el])——设置方位角和仰角；

view([x,y,z])——设置视角用笛卡尔直角坐标，它的幅值是忽略的；

view(2)——设置二维视角，默认值为 az = 0°，el = 90°；

view(3)——设置三维立体图的视角，默认值为 az = －37.5°，el = 30°；

view(T)——视角设置依照转换矩阵 T，它是 4×4 的矩阵，作为视角转换，它由 viewmtx（视角矩阵）函数产生；

[az,el] = view ——返回当前的方位角和仰角；

T = view——返回当前 4×4 转换矩阵。

除了用上述 view 命令外，视角的控制还可以通过当前的图形的编辑栏（edit），点击图形属性编辑器，选择轴特性，再选择 viewpoint 选项卡，来实现视角的改变。

【例 5-41】 默认视角下的带有缺口的衰减正弦波。

解：

```
≫[X,Y] = meshgrid(-4:0.2:4);                              %设置网格矩阵
≫Z = exp(-(0.15*X).^2-Y.^2).*sin(X)./(X+eps);            %计算矩阵 Z
≫surfc(X,Y,Z)                                             %绘制带有等高线的网格图，如图
                                                            5-41 所示
```

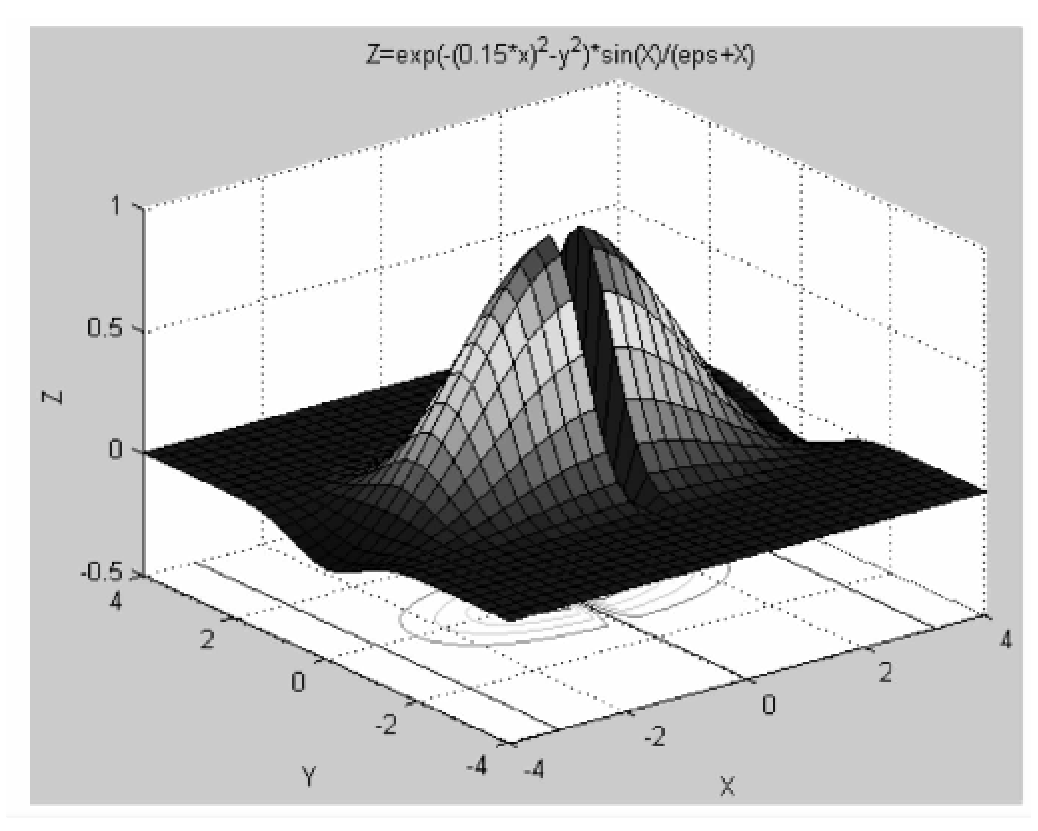


图 5-41　view（3）时的有缺口的衰减正弦波

```
≫title('Z = exp(-(0.15*X).^2-Y.^2).*sin(X).*sin(Y)的表面图')
                                                            %设置标题
≫xlabel('X')                                               %设置坐标标签
≫ylabel('Y')
≫zlabel('Z')
```

图 5-41 是选用默认的视角。图 5-42 则将视角改变为 10.5°和 30°。在命令窗口输入 view 命令即显示当前图形（图 5-42）的视角和转换矩阵

```
≫[az,el] = view
az =
    10.5000
el =
    30
≫T = view
T =
```

0.9833	0.1822	0.0000	-0.5827
-0.0911	0.4916	0.8660	-0.6333
-0.1578	0.8515	-0.5000	8.5634
0	0	0	1.0000

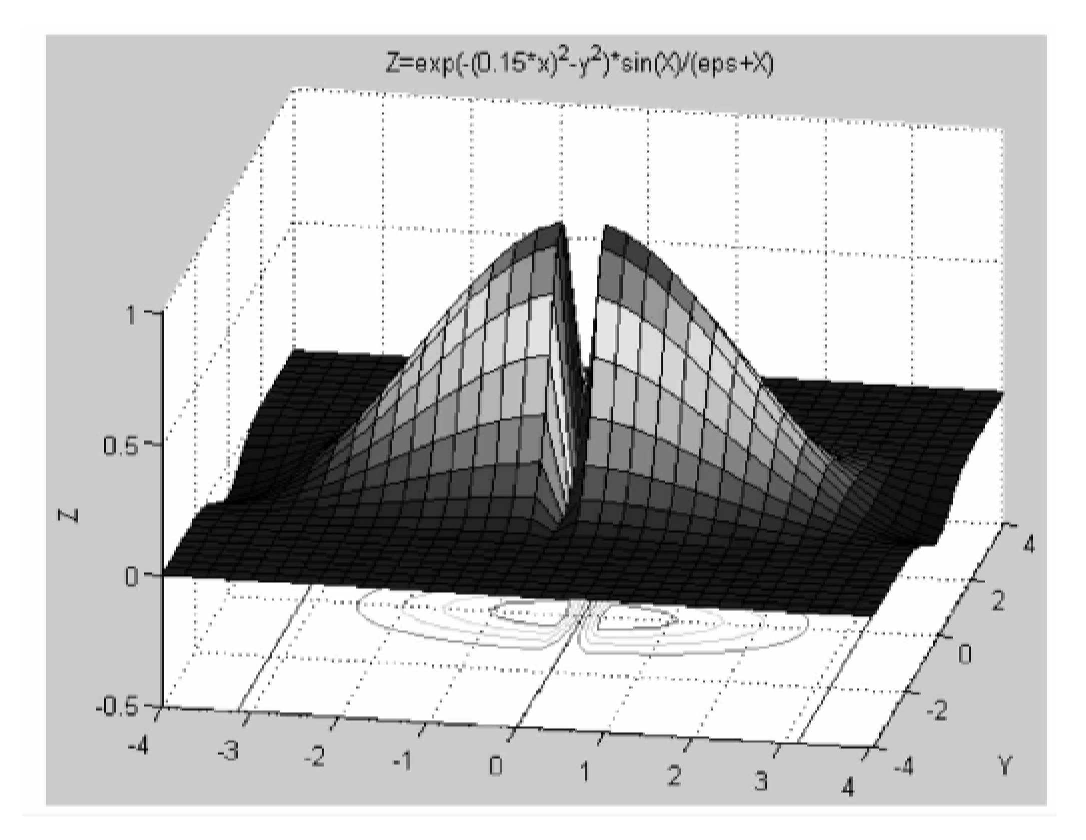


图 5-42　视角为 viewP（10.5，30）**时的有缺口的衰减正弦波**

当前图形的颜色设置和获取是由 colormap（色图）函数来实现的。Colormap 的书写格式为

colormap(map)

colormap('default')

cmap = colormap

第一种格式为图形的颜色设置，颜色的分配是由矩阵 map 决定。map 是 m×3 的矩阵，它的每一元素是 0.0~1.0 之间的实数，它的每一行的 RGB（色彩）向量，定义一种颜色。图形有 k 行，colormap 就确定 k 行颜色，map(k,:) = [r(k),g(k),b(k)]指定红、绿、蓝 3 色的深度。

第二种格式是默认的设置。图 5-41、图 5-42 的图形色彩就是默认设置的色彩。

第三种格式是提取当前图形的色彩矩阵。图 5-41、图 5-42 的图形色彩矩阵如下。

```
≫colormap
ans =
```

0	0	0.5625
0	0	0.6250
0	0	0.6875

0	0	0.7500
0	0	0.8125
0	0	0.8750
0	0	0.9375
0	0	1.0000
0	0.0625	1.0000
0	0.1250	1.0000
0	0.1875	1.0000
0	0.2500	1.0000
0	0.3125	1.0000
0	0.3750	1.0000
0	0.4375	1.0000
0	0.5000	1.0000
0	0.5625	1.0000
0	0.6250	1.0000
0	0.6875	1.0000
0	0.7500	1.0000
0	0.8125	1.0000
0	0.8750	1.0000
0	0.9375	1.0000
0	1.0000	1.0000
0.0625	1.0000	0.9375
0.1250	1.0000	0.8750
0.1875	1.0000	0.8125
0.2500	1.0000	0.7500
0.3125	1.0000	0.6875
0.3750	1.0000	0.6250
0.4375	1.0000	0.5625
0.5000	1.0000	0.5000
0.5625	1.0000	0.4375
0.6250	1.0000	0.3750
0.6875	1.0000	0.3125
0.7500	1.0000	0.2500
0.8125	1.0000	0.1875
0.8750	1.0000	0.1250
0.9375	1.0000	0.0625
1.0000	1.0000	0
1.0000	0.9375	0

```
    1.0000    0.8750         0
    1.0000    0.8125         0
    1.0000    0.7500         0
    1.0000    0.6875         0
    1.0000    0.6250         0
    1.0000    0.5625         0
    1.0000    0.5000         0
    1.0000    0.4375         0
    1.0000    0.3750         0
    1.0000    0.3125         0
    1.0000    0.2500         0
    1.0000    0.1875         0
    1.0000    0.1250         0
    1.0000    0.0625         0
    1.0000         0         0
    0.9375         0         0
    0.8750         0         0
    0.8125         0         0
    0.7500         0         0
    0.6875         0         0
    0.6250         0         0
    0.5625         0         0
    0.5000         0         0
```

由于色彩矩阵的设置是比较费时的，因此 MATLAB 设置了一些色彩函数，用户选择了色彩函数名，把它添加到 colormap 后的圆括号内执行即可。MATLAB 支持以下几种色彩函数。

autumn——由红色向桔黄色、黄色平滑过渡；

bone——灰色的色图，具有较深的蓝色成分；

colorcube——包含 RGB 颜色空间中尽可能多的有规则色彩，提供灰色，纯红，纯绿，纯蓝多步间隔的颜色；

cool——由青色和洋红组成阴暗的颜色，它在青色与洋红间平滑过渡；

copper——在黑色与古铜色间平滑过渡；

flag——由红色，白色，蓝色和黑色组成。这种色图完全改变了颜色随矩阵下标的增加而改变；

gray——返回线性变化的灰色图；

hot——在黑色、红色、桔红色、黄色和白色之间平滑过渡；

hsv——改变色彩饱和度模型中的色彩成分。颜色从红色开始，然后黄色、绿色、青色、蓝色、洋红，最后返回到红色。这种色图特别适用于显示周期性函数；

jet——在蓝色、青色、黄色、桔红色、红色之间过渡。它是 hsv 色图的变种。Jet 色图是与天体物理学中模拟喷射气流相关联的；

lines——由坐标轴中的 ColorOrder（色彩序列）属性生成的色图和灰暗色组成；

pink——包含品红的柔和暗色。这颜色提供棕褐色灰暗色彩格调；

prism——重复红色、桔红色、黄色、绿色、蓝色和紫色 6 种颜色；

spring——由颜色深浅的洋红和黄色组成；

summer——由颜色深浅的绿色和黄色组成；

white——白色图；

winter——由颜色深浅的蓝色和绿色组成。

读者可以在绘制三维立体图时，通过色图的设置来体会各种色彩函数的特性。

第 5 章习题

5-1　在 $x=[-4,4]$，绘制 $y=\frac{\sin^2 x}{1+x^2}$ 的线性图。

5-2　绘制发射角 $a=30$、45、75 度时的抛物线图，$\boldsymbol{x}=[0\ 1200]$，$v=100$，$g=9.8$，抛物线方程式为

$$y = x\tan a - \frac{gx^2}{2v^2\cos^2 a}$$

5-3　用极坐标和直角坐标分别绘图函数 polar 绘制绳结线 $r=-\cos(2x)\sec x$，$\boldsymbol{x}=[0,2\pi]$。

5-4　在单位于圆内绘制五角星图。

5-5　用 5 阶魔方矩阵绘制条形图。

5-6　某 3 口家庭，月收入 5500 元。开支如下：

(1) 住房还贷	2200
(2) 饮食费用	1500
(3) 文教费	600
(4) 医药费用（平均数）	200
(5) 交通费	300
(6) 储蓄	700

试绘制饼图，并把支出最大部分从饼图中分离出来。

5-7　利用柱形图函数 cylinder 和表面图函数 surf 生成花瓶立体图。图形式样可自由选取。

第 6 章　多项式、插值和曲线拟合

多项式在数学中占有重要的地位。多项式的四则运算，多项式求根，多项式的求值，任意函数展开成 Taylor 级数，多项式与伴随矩阵的转换，散点图的曲线拟合，测试数据的多项式回归，分式多项式展开成部分分式，传递函数的表示等都要用到多项式。MATLAB 为多项式运算提供了多种运算函数，其常用的有关多项式函数见表 6-1。

表 6-1　常用多项式运算函数

函　数　名	说　　明
conv	多项式乘法
deconv	多项式除法
poly	由多项式的根构造系数多项式
polyval	求多项式的值
polyvalm	求矩阵多项式的值
sym2poly	山符号多项式构成系数多项式
poly2sym	由系数多项式构成符号多项式
polyder	求符号多项式的微商表达式
polyint	求符号多项式的积分表达式
polyfit	多项式的曲线拟合
polytool	多项式的曲线拟合工具
compan	山系数多项式生成伴随矩阵
polyeig	多项式的特征值
residue	分式多项式展开成部分分式
roots	求多项式的根
Taylortool	Taylor 级数工具

插值是在已知数据中寻找估计值的过程。它在数据处理和图形处理中占有重要的地位。举一个简单的例子，若已知向量 $\boldsymbol{X}$ = [1 2 3 4 5]，对应的向量 $\boldsymbol{Y}$ = [1 4 9 16 25]，那么对于向量 $\boldsymbol{X}_1$ = [1 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0]，则对应的向量 $\boldsymbol{Y}_1$ 值应为多少，这就是插值的问题。插值可分为一维插值、二维插值和多维插值，在一维插值中，又分为线性插值和样条插值。

对于曲线拟合，它是不同于插值运算，它是从试验数据中归纳为可以用已知函数组合来描述的方程式，而这方程式是最佳逼近试验数据。这里所指的最佳逼近，是指数据实际值与逼近方程式的计算值的误差二乘方之和最小。MATLAB 常用插入函数见表 6-2。

表 6-2　常用插入函数表

函　数　名	说　　明
interp1	一维插值
inpterp2	二维插值
spline	立体样条插值
interpft	一维快速傅里叶插值

6.1 多项式的表示

在数学上，多项式通常用变量的降幂形式书写。例如多项式

$$p(x) = x^5 + 2x^4 + 5x^3 + 3x + 13 \tag{6-1}$$

在 MATLAB 中，多项式（6-1）是用系数向量 P 来表示，即

$$P = [1\ 2\ 5\ 0\ 3\ 13] \tag{6-2}$$

在向量 P 中依次写入式（6-1）中的多项式系数，对于缺项，则用零元素补充。多项式在采用向量表示法后，使多项式的运算变得十分简单，它与多项式变量的符号无关。

多项式的阶次是依照系数向量的长度减 1 而得。即多项式的阶次

$$n = \mathrm{size}(P) - 1 \tag{6-3}$$

式（6-2）也可以由 MATLAB 自动改写。方法如下，在 MATLAB 命令窗口输入如下程序：

```
≫syms x                                  %设置符号变量 x
≫P = x^5 + 2* x^4 + 5* x^3 + 13 + x;     %写入多项式表达式，可以不按降幂排列
≫P = sym2poly(P)                         % 调用符号表达式转换系数多项式函数 sym2poly
                                           则得系数多项式是按降幂排列

P =
    1    2    5    0    1    13
≫poly2sym(P)                             %系数向量 P 的逆变换函数 poly2sym，将向量 P 转
                                           换成符号表达式，即通常的书写形式

ans =
x^5 + 2* x^4 + 5* x^3 + x + 13
```

6.2 多项式的根

多项式求根，对于高阶方程式是比较复杂、繁琐的工作。但采用了 MATLAB 的求多项式根的函数 roots 后，则求解多项式的根变成轻而易举的事。多项式求根函数的书写格式为

$$R = \mathrm{roots}(p) \tag{6-4}$$

式中，向量 R 返回一个有 n 个元素的列向量，它的每一个元素代表多项式的一个根。式中 p 为多项式的系数向量。假若 p 是 n + 1 个元素的向量，则它的多项式为

$$p(1) * x\hat{}n + p(2) * x\hat{}(n-1) + \cdots + p(n) * x + p(n+1) \tag{6-5}$$

下面举例说明它的应用。

【例 6-1】 已知 5 阶多项式的系数向量为 $\boldsymbol{P} = [1\ 8\ 28\ 58\ 67\ 30]$，求多项式的根。

解：

在 MATLAB 窗口输入如下程序：

```
≫P = [1 8 28 58 67 30]                   %输入多项式系数向量
P =
    1    8    28    58    67    30
≫R = roots(P)                            %计算多项式的根，其中有 3 个负实根，1 对具有
                                           负实部的复根
```

```
R =
    -1.0000 + 2.0000i
    -1.0000 - 2.0000i
    -3.0000
    -2.0000
    -1.0000
```

【例 6-2】 已知多项式为 $s^5-6s^4+36s^3-72s^2+104s-55=0$，求多项式的根。

解：

在 MATLAB 窗口输入如下程序：

```
>>syms s                                      %设置符号变量 s
>>P1 = s^5 - 6* s^4 + 36* s^3 - 72* s^2 + 104* s - 55;   %输入5阶多项式
>>P1 = sym2poly(P1)                           %转换符号多项式为系数多项式
   P1 =
        1     -6     36     -72     104     -55
>>R1 = roots(P1)                              %计算多项式的根，有1个正实根，
                                                2对具有正实部的复根
R1 =
   1.8284 + 4.4724i
   1.8284 - 4.4724i
   0.7550 + 1.5025i
   0.7550 - 1.5025i
   0.8332
```

与多项式求根函数 roots 可逆的函数为 poly，它是由已知多项式的根 r1，r2，…，rn 去构造系数多项式。它的书写格式为

$$P = \text{poly}(R) \tag{6-6}$$

式中 R = [r1 r2…rn]，P 返回一个多项式系数向量。它是由向量[1 - r1]，[1 - r2]，…，[1 - rn]卷积而成。

若 R 为 n×n 的方阵，则 P 也返回多项式系数向量，这向量是由

$$\text{sym2poly}(\text{det}(\text{lambda}^* \text{eye}(\text{size}(R)) - R)) \tag{6-7}$$

表示出。(6-7) 式中 lambda 为符号变量。向量 P 称为矩阵 R 的特征多项式。下面举例说明，如何由多项式的根创建系数多项式。

【例 6-3】 已知多项式的根为 -1、-2、-3、-4+3j、-4-3j，求多项式。

解：

```
>>R = [-1  -2  -3  -4 + 3* j  -4 - 3* j]        %输入多项式根的向量
R =
    -1.0000     -2.0000    -3.0000     -4.0000 + 3.0000i   -4.0000 - 3.0000i
>>P = poly(R)                                  %构造多项式的系数向量
P =
    1     14     84     244     323     150
```

```
>>P = poly2sym(P)                    %转换系数多项式为符号多项式
P =
x^5 + 14*x^4 + 84*x^3 + 244*x^2 + 323*x + 150
```

【例 6-4】 已知 4 阶 Pascal 矩阵 **A**，求其多项式向量，多项式的根，并与矩阵 A 的特征值相比较。

解：

```
>>A = pascal(4)                      %输入 4 阶 Pascal 矩阵 A
A =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20

>>P = poly(A)                        %计算矩阵 A 的特征多项式 P
P =
    1.0000   -29.0000   72.0000   -29.0000   1.0000
>>R = roots(P)                       %计算系数多项式 P 的根
R =
    26.3047
     2.2034
     0.4538
     0.0380
    >>eig(A)                         %计算矩阵 A 的特征值，与向量 R 相同，但排列次序不同
ans =
     0.0380
     0.4538
     2.2034
    26.3047
```

6.3 多项式的乘除

假若有两个多项式系数向量 $\boldsymbol{U}$，$\boldsymbol{V}$，其长度分别为 m、n，则其乘积向量为 $\boldsymbol{W}$，它的长度为 $m+n-1$，向量 **W** 中的第 k 个元素为

$$W(k)=\sum u(j)v(k+1-j), j \text{ 从 } \min(k,m) \text{ 到 } \max(1,k+1-n) \tag{6-8}$$

式中，$u(j)$，$v(k+1-j)$ 分别为向量 $\boldsymbol{U}$，$\boldsymbol{V}$ 的元素。在 MATLAB 中多项式乘法函数为 conv，其书写格式为

$$\mathrm{W = conv(U,V)} \tag{6-9}$$

【例 6-5】 已知多项式系数向量 $\boldsymbol{U}=[1\ 3\ 5\ 7\ 5\ 3\ 1]$、$\boldsymbol{V}=[1\ 4\ 2]$，求这两个多项式的乘积的符号表达式。

解：

```
>>U = [1 3 5 7 5 3 1];                %输入多项系数向量 U
>>V = [1 4 2];                        %输入多项系数向量 V
>>W = conv(U,V)                       %计算 U、V 乘积的系数向量
W =
     1     7    19    33    43    37    23    10     2
>>poly2sym(W)                         %转换多项式系数向量为符号表达式
ans =
x^8 + 7* x^7 + 19* x^6 + 33* x^5 + 43* x^4 + 37* x^3 + 23* x^2 + 10* x + 2
```

多项式的除法是乘法的逆运算，在 MATLAB 中多项式的除法的书写格式为

$$[Q,R] = \mathrm{deconv}(A,B) \tag{6-10}$$

式中，A 为被除式系数向量，B 为除式系数向量，Q 为商式系数向量，R 为余式系数向量。

deconv 为多项式除法函数。下面举例予以说明。

【例 6-6】 已知多项式系数向量 ***A*** = [1 5 10 10 5 1]，多项式系数向量 ***B*** = [1 3 2]，求 ***A/B*** 的商和余数多项式。

解：

```
>>A = [1 5 10 10 5 1];               %输入系数向量 A
>>B = [1 3 2];                        %输入系数向量 B
>>[Q,R] = deconv(A,B)                 %计算 A、B 相除后的商和余数表达式
Q =                                   %商表达式
     1     2     2     0
R =                                   %余数表达式
     0     0     0     0     1     1
```

【例 6-7】 已知多项式系数向量 ***A*** = [1 6 17 32 44 44 32 17 6 1]，多项式系数向量 ***B*** = [1 3 3 1]，求 ***A/B*** 的商和余数多项式。

解：

```
>>A = [1 6 17 32 44 44 32 17 6 1];%输入向量 A
>>B = [1 3 3 1];                      %输入向量 B
>>[Q,R] = deconv(A,B)                 %计算 A、B 相除后的商和余数表达式
Q =                                   %商的系数向量
     1     3     5     7     5     3     1
R =                                   %余数为零，除尽
     0     0     0     0     0     0     0     0     0     0
```

6.4 多项式的值

在寻找多项式的极大值、极小值和多项式的根时，需要计算多项式的值。在 MATLAB 中计算多项式的值的书写格式如下：

$$Y = \mathrm{polyval}(P,X) \tag{6-11}$$

式中，P 为多项式系数向量或矩阵，X 为向量或矩阵。Y 为式（6-11）的输出向量或矩阵，它对应输出向量为

$$Y = P(1)*X.\hat{}N + P(2)*X.\hat{}(N-1) + \cdots + P(N)*.X + P(N+1)*I \tag{6-12}$$

式中，N 为系数多项式 P 的长度减 1，即 N = size(P) - 1。

式（6-12）计算多项式的值是考虑数组乘方。若考虑矩阵乘方时 X 必需为方阵，并且书写格式改为

$$Y = \text{polyvalm}(P, X) \tag{6-13}$$

式中，polyvalm 为多项式矩阵求值函数。

【例 6-8】 已知多项式的系数向量为 $\boldsymbol{P} = [1\ 3\ 3\ 1]$，当 $\boldsymbol{X} = [1\ 2\ 3\ 4\ 5\ 6\ 7]$ 及 $\boldsymbol{X}_1$ 为 4 阶 pascal 矩阵，分别求数组多项式的值 $\boldsymbol{Y}$ 和 $\boldsymbol{Y}_1$。

解：

```
≫P = [1 3 3 1];                    % 输入多项式系数向量 P
≫X = [1:7]                         % 输入自变量向量 X
X =
     1     2     3     4     5     6     7
≫X1 = pascal(4)                    % 输入自变量矩阵 X1
X1 =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
≫Y = polyval(P,X)                  % 计算多项式的向量 Y
Y =
     8    27    64   125   216   343   512
≫Y1 = polyval(P,X1)                % 计算多项式的矩阵 Y1
Y1 =

        8        8        8        8
        8       27       64      125
        8       64      343     1331
        8      125     1331     9261
```

【例 6-9】 已知多项式系数向量 $\boldsymbol{P} = [1\ 3\ 3\ 1]$，$\boldsymbol{X}$ 为 4 阶 pascal 矩阵，求矩阵多项式的值。

解：

```
≫P = [1 3 3 1];                    % 输入多项式系数向量 P
≫X = pascal(4)                     % 输入自变量矩阵 X
X =

     1     1     1     1
     1     2     3     4
     1     3     6    10
```

```
     1      4     10     20
>>Y = polyvalm(P,X)             %计算矩阵多项式的值
Y =
      85     257     567    1052
     257     838    1884    3529
     567    1884    4278    8049
    1052    3529    8049   15193
```

6.5 多项式的微分

在 MATLAB 中多项式微分书写格式为

$$k = polyder(p) \tag{6-14}$$

$$k = polyder(a,b) \tag{6-15}$$

$$[q,d] = polyder(b,a) \tag{6-16}$$

式中，函数 polyder 用来计算多项式、多项式的乘积和分式多项式的微商。式（6-14）~式（6-16）中的操作数 a、b 和 p 是系数向量。它的元素是依降幂排列的。

式（6-14）返回系数多项式 p 的微商。式（6-15）返回系数多项式 a、b 乘积的微商。式（6-16）则返回分式多项式 b/a 的微商，而 q 表示分子系数多项式，d 则表示分母系数多项式。下面举例予以说明。

【例 6-10】 已知多项式 $p = x^4 - 3x^2 + 5x - 7$，求它的微商表达式。

解：

```
>>syms x                        %设符号变量
>>p = x^4 - 3*x^2 + 5*x - 7;    %输入多项式 p
>>p = sym2poly(p)               %转换多项式为系数多项式 p
p =
    1    0   -3    5   -7
>>k = polyder(p)                %计算多项式的微商表达式 k
k =
    4    0   -6    5
>>k = poly2sym(k)               %将表达式 k 转换成符号表达式
k =
    4*x^3 - 6*x + 5
```

【例 6-11】 已知多项式 $a = x^4 - 3x^2 + 5x - 7$，$b = x^2 + 3x + 2$，求 ab 的微商表达式。

解：

```
>>syms x                        %设符号变量
>>a = x^4 - 3*x^2 + 5*x - 7;    %输入多项式 a
>>b = x^2 + 3*x + 2;            %输入多项式 b
>>a = sym2poly(a)               %转换符号多项式为系数多项式 a
a =
```

```
    1    0    -3    5    -7
>>b = sym2poly(b)                         %转换符号多项式为系数多项式 b
b =
    1    3    2
>>k = polyder(a,b)                        %计算多项式 a、b 乘积的微商
k =

    6    15    -4    -12    4    -11
>>k = poly2sym(k)                         %转换多项式 k 为符号表达式
k =
6*x^5+15*x^4-4*x^3-12*x^2+4*x-11
```

【例 6-12】 已知多项式 $b=x^5+5x^4+9x^3+9x^2+5x+2$，$a=x^2+3x+1$，求 b/a 的微商表达式。

解：

```
>>syms x                                  %设置符号变量
>>a = x^2+3*x+1;                          %输入除式多项式 a
>>b = x^5+5*x^4+9*x^3+9*x^2+5*x+2;        %输入被除式多项式 b
>>a = sym2poly(a)                         %转换成系数多项式
a =
    1    3    1
>>b = sym2poly(b)                         %转换成系数多项式
b =
    1    5    9    9    5    2
>>[q,d] = polyder(b,a)                    %计算 b/a 的微商
q =                                       %分子多项式
    3    22    59    74    49    14    -1
d =                                       %分母多项式
    1    6    11    6    1
>>[Q,D] = deconv(q,d)                     % q、d 相除，注意大写的 Q、D 与小写的 q、
                                            d 为不同的变量
Q =                                       %相除后得的商
    3    4    2
D =                                       %相除后的余项
    0    0    0    0    0    -2    -3
>>Q = poly2sym(Q)                         %转换成符号表达式
Q =
3*x^2+4*x+2
>>D = poly2sym(D)                         %转换成符号表达式
D =
```

```
-2*x-3
>>d=poly2sym(d)                          %转换成符号表达式
d =
x^4+6*x^3+11*x^2+6*x+1
>>Dab=Q+D/d                              %多项式 b、a 相除后的微商表达式
Dab =
3*x^2+4*x+2+(-2*x-3)/(x^4+6*x^3+11*x^2+6*x+1)
```

【例 6-13】 今有一块 3×5 的矩形薄钢板，拟在 4 个角剪去边长为 x 的正方形，围成高为 x，边长各为 $3-x$ 和 $5-x$ 的立方体，问 x 应为多少，才能使这立方体体积最大。

解：

```
>>syms x                                 %设置符号变量
>>l=[-1 5];                              %输入边长向量，边长为 (5-x)
>>b=[-1 3];                              %输入边宽向量，边宽为 (3-x)
>>h=[1 0];                               %输入高度向量，高为 (x +0)
>>V=conv(conv(l,b),h)                    %立方体体积多项式系数向量
V =
     1    -8    15     0
>>k=polyder(V)                           %取多项式 V 的微商 k
k =
     3    -16    15
>>k=poly2sym(k)                          %转换成符号表达式 k
k =
3*x^2-16*x+15
>>solve(k)                               %解微商表达式
ans =
[8/3+1/3*19^(1/2)]
[8/3-1/3*19^(1/2)]
>>x=eval(ans)                            %将字符串转换成数字
x =                                      %去除增根，选取x(2)为最佳值
     4.1196
     1.2137
>>Vmax=polyval(V, x(2))                  %计算最佳值时的容积
Vmax =
     8.2088
>>x(2)*(3-x(2))*(5-x(2))                 %核对
ans =
     8.2088
>>1.2*(3-1.2)*(5-1.2)                    %偏离最佳值时，取 x=1.2 时的容积小于 Vmax
ans =
```

8.2080

6.6 多项式的积分

在 MATLAB 中多项式积分书写格式为

$$y = \text{polyint}(p, k) \tag{6-17}$$

$$y = \text{polyint}(p) \tag{6-18}$$

式中，函数 polyint 为计算多项式的不定积分，其中 p、y 为多项式的系数向量。式（6-17）考虑积分常数 k，式（6-18）则不计积分常数，并认为 k = 0。

【例 6-14】 已知 $p = x^3 + 2x^2 + 5x + 7$，求其积分表达式，考虑积分常数为零。

解：

```
≫syms x                              %设置符号变量
≫p = x^3 + 2* x^2 + 5* x + 7;        %输入被积表达式
≫p = sym2poly(p)                     %转换成系数向量
p =
    1     2      5     7
≫y = polyint(p)                      %对多项式进行积分
y =
    0.2500    0.6667    2.5000    7.0000    0
≫y = poly2sym(y)                     %将积分后的系数向量，转换成符号表达式
y =
1/4* x^4 + 2/3* x^3 + 5/2* x^2 + 7* x
```

注意，多项式必须要化成系数多项式后才能使用 polyint 函数，否将导致错误。例如下面的运算是错误的。

```
≫syms x                              %设置符号变量
≫y = x^3 - 3* x;                     %输入多项式
≫polyint(y)                          %求多项式积分
ans =                                %错误结果
[x^3 - 3* x,          0]
≫int(y)                              %正确的符号积分
ans =
1/4* x^4 - 3/2* x^2
```

6.7 分子与分母多项式的提取

分式多项式中提取分子或分母在运算中是有用的。它的书写格式为

$$[n, d] = \text{numden}(A) \tag{6-19}$$

式中，n 为分子多项式，d 为分母多项式，A 为分式多项式。numden 为分子、分母多项式提取函数。当 A 为数字分式时，该函数仍然有效。

【例 6-15】 已知数字分式2/{3 + [2/(3 + 2/3)]}，求分子及分母。

解：

```
≫[n,d]=numden(sym(2/(3+2/(3+2/3))))          %将分式转换成符号数字，再提取
                                               分子与分母
n=                                            %分子
22
d=                                            %分母
39
≫n/d
ans=
22/39
≫eval(ans)                                    %分式的值
ans=
0.5641
```

【例 6-16】 已知分式多项式 $y=3/(x+5)+(x+8)/(x^2+6x+5)$，求通分后的分子与分母多项式。

解：

```
≫syms x                                       %设符号变量
≫y=3/(x+5)+(x+8)/(x^2+6*x+5);                 %输入符号多项式
≫[n,d]=numden(y)                              %提取分子与分母多项式
n=
4*x^2+31*x+55
d=
(x+5)*(x^2+6*x+5)
≫n=(x+5)*(4*x+11);                            %对分子分解因式
≫[n,d]=numden(n/d)                            %再次提取分子分母
n=
4*x+11
d=
x^2+6*x+5
```

6.8 分式多项式转换成部分分式

分式多项式转换成部分分式，在自动调节理论中占有重要的地位。因为它可以将复杂的传递函数转换成简单的传递函数之和。由此可解出在不同输入作用下的瞬态响应。

在 MATLAB 中，对分式多项式与部分分式表达式之间进行转换的函数为 residue，它的书写格式为

$$[r,p,k]=\text{residue}(b,a) \tag{6-20}$$

$$[b,a]=\text{residue}(r,p,k) \tag{6-21}$$

式中，r 为留数的列向量；p 为极点的列向量；k 为直接项的行向量；b 为分子多项式b(s)的系数向量，以降幂排列；a 为分母多项式a(s)的系数向量，以降幂排列。

有关留数详情，请查阅数学手册。若分式多项式 $b(s)/a(s)$ 有一阶极点 pi，则在此点的

留数为

$$r_i = \mathrm{Res}(b(p_i)/a(p_i)) = \lim_{s=p_i}\left((s-\mathrm{pi})\frac{b(s)}{a(s)}\right) \tag{6-22}$$

其中，

$$b(s) = b_1 s^m + b_2 s^{m-1} + \cdots + b_{m+1} \tag{6-23}$$

$$a(s) = a_1 s^n + a_2 s^{n-1} + \cdots + a_{n+1} \tag{6-24}$$

式中，s 为运算变量

分式多项式与部分分式的关系为

$$b(s)/a(s) = r_1/(s-p_1) + r_2/(s-p_2) + \cdots + r_n/(s-p_n) + k(s) \tag{6-25}$$

下面举例说明它的应用。

【例 6-17】 已知分式多项式如下，请展开成部分分式。

$$F(x) = \frac{x+2}{(x+1)(x+3)(x+5)}$$

解：

```
>>a1 = [1 1];                    %设置分母项(x + 1)
>>a2 = [1 3];                    %设置分母项(x + 3)
>>a3 = [1 5];                    %设置分母项(x + 5)
>>B = [1 2];                     %设置分子项 x + 2
>>A = conv(conv(a1,a2),a3)       %用多项式乘法 conv，计算分母多项式
A =
    1     9     23     15
>>[R,P,K] = residue(B,A)         %展开成部分分式，其中 R 为留数向量，P 为极点
                                   向量，K 为直接项向量
R =                              %部分分式的分子
   -0.3750
    0.2500
    0.1250
P =                              %部分分式分母的极点
   -5.0000
   -3.0000
   -1.0000
K =                              %直接项为空矩阵
   []
```

由此可得展开的部分分式如下：

$$F(x) = \frac{x+2}{(x+1)(x+3)(x+5)}$$

$$= \frac{0.125}{(x+1)} + \frac{0.25}{(x+3)} - \frac{0.375}{(x+5)}$$

【例 6-18】 已知分式多项式如下，请展开成部分分式。

$$F(x) = \frac{x^2+2x+5}{(x+5)(x^2+2x+4)}$$

解：

```
>>B = [1,2,5];                               %设置分子多项式系数向量
>>A = conv([1,5],[1,2,4])                    %设置分母多项式系数向量
A =
    1     7     14     20
>>[r,p,k] = residue(B,A)                     %展开成部分分式
r =                                          %部分分式的分子
1.0526
    -0.0263 - 0.0608i
    -0.0263 + 0.0608i
p =                                          %部分分式的极点
    -5.0000
    -1.0000 + 1.7321i
    -1.0000 - 1.7321i
k =                                          %直接项为空矩阵
    []
>>a1 = [1, -p(1)];                           %部分分式第1项的分母系数向量
>>a2 = [1, -p(2)];                           %部分分式第2项的分母系数向量
>>a3 = [1, -p(3)];                           %部分分式第3项的分母系数向量
>>b = conv(r(2),a3) + conv(r(3),a2)          %部分分式第2、3项通分后的分子系数向量
b =
    -0.0526    0.1579
```

由此可得展开的部分分式如下：

$$
\begin{aligned}
F(x) &= \frac{x^2+2x+5}{(x+5)(x^2+2x+4)} \\
&= \frac{1.0526}{x+5} + \frac{-0.0263-0.0608\mathrm{i}}{x+1-1.7321\mathrm{i}} + \frac{-0.0263+0.0608\mathrm{i}}{x+1+1.7321\mathrm{i}} \\
&= \frac{1.0526}{x+5} + \frac{-0.0526x+0.1579}{x^2+2x+4}
\end{aligned}
$$

【例 6-19】 已知留数向量 $\boldsymbol{r} = [1\ 2\ 3]$，极点向量 $\boldsymbol{p} = [-1\ \ -2\ \ -3]$，直接项 $k=0$，求分式多项式 $b(s)/a(s)$。

解：

```
>>r = [1;2;3];                               %输入留数向量
>>p = [-1; -2; -3];                          %输入极点向量
>>k = 0;                                     %设 k = 0
>>[b,a] = residue(r,p,k)                     %计算分式多项式
b =                                          %分子多项式系数
    6     22     18
a =                                          %分母多项式系数
```

```
     1        6        11       6
>>b = poly2sym(b,'s')                       %转换成分子符号表达式，即 b (s)
b =
6* s^2 + 22* s + 18
>>a = poly2sym(a,'s')                       %转换成分母符号表达式，即 a (s)
a =
s^3 + 6* s^2 + 11* s + 6
```

【例 6-20】 已知分式多项式的分子 $\boldsymbol{b}$ = [1 15 31 －1]，分母多项式 $\boldsymbol{a}$ = [1 －8 14 8 －15]，求分解为部分分式。

解：

```
>>b = [1 -15 31 -1];                        %输入分子多项式系数
>>a = [1 -8 14 8 -15];                      %输入分母多项式系数
>>[r,p,k] = residue(b,a)                    %计算留数和极点
r =                                         %与极点相关的留数值
    -2.0000
     1.0000
     1.0000
     1.0000
p =                                         %极点的值
     5.0000
     3.0000
     1.0000
    -1.0000
k =                                         %直接项是空的
     []
```

所以 $b(s)/a(s) = 1/(s-3)+1/(s-1)+1/(s+1)-2/(s-5)$

6.9 多项式与伴随矩阵

伴随矩阵在描述自动控制系统的状态方程是有用的。它与多项式有密切的联系。伴随矩阵的书写格式为

$$A = \text{compan}(V) \tag{6-26}$$

式中，V 为多项式系数向量，A 为伴随矩阵。A 矩阵的第一行是 － V(2:n)/V(1)，它的左下角为(n－1)阶的单位矩阵，其余元素为零。A 的特征值为系数多项式的根。现举例如下：

【例 6-21】 已知系数多项式 $\boldsymbol{V}$ = [1 0 －19 30]，求多项式的根，伴随矩阵 $\boldsymbol{A}$，矩阵 $\boldsymbol{A}$ 的特征根。

解：

```
>>V = [1 0 -19 30]                  %输入多项式向量 V
V =
     1        0       -19       30
```

```
≫roots(V)                          %多项式根
ans =
    -5.0000
    3.0000
    2.0000
≫A = compan(V)                     %创建伴随矩阵 A
A =
       0      19     -30
       1       0       0
       0       1       0
≫eig(A)                            %矩阵 A 的特征值，与多项式 V 的根完全一致
ans =
       -5.0000
        3.0000
        2.0000
```

6.10 多项式的曲线拟合

多项式的曲线拟合，是用一个多项式表达式去拟合一组试验数据，从中寻找规律，为创建经验公式，或为数据预测准备条件。在 MATLAB 中，是用 polyfit 函数作为多项式曲线拟合的。它的书写格式为

$$p = \mathrm{polyfit}(x, y, n) \tag{6-27}$$

$$[p, S] = \mathrm{polyfit}(x, y, n) \tag{6-28}$$

$$[p, S, mu] = \mathrm{polyfit}(x, y, n) \tag{6-29}$$

式中，x、y 为数据向量，p 为多项式系数向量，n 为多项式阶次。S 为描述拟合误差的结构数组。mu = [μ1, μ2]，μ1 = mean(x)，μ2 = std(x)。

式（6-27）是用 n 阶多项式

$$p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{(n+1)} \tag{6-30}$$

去拟合试验数据，即 $p(x_i)$ 的值在满足最小二乘的条件下去适合 y_i。一般情况下，当增大 n，则计算误差就减小，多项式函数越接近实际值。但计算工作量亦相应增加。下面举例予以说明。

【例 6-22】 已知正弦函数 $y = \sin(x)$，取值向量 $\boldsymbol{x} = [0\ \ \pi/18\ \ \pi/9\ \ \pi/6\ \ \cdots\ \ \pi/2]$，向量 $\boldsymbol{y} = [\sin(0)\sin(\pi/18)\cdots\sin(\pi/2)]$，用 3 阶多项式来拟合正弦曲线，求多项式系数。

解：

```
≫x = [0:pi/18:pi/2]';               %输入 x 向量
≫y = sin(x);                        %计算输出向量
≫table = [x,y]                      %作设定值正弦函数表
table =
```

```
         0         0
    0.1745    0.1736
    0.3491    0.3420
    0.5236    0.5000
    0.6981    0.6428
    0.8727    0.7660
    1.0472    0.8660
    1.2217    0.9397
    1.3963    0.9848
    1.5708    1.0000
≫[p,s] = polyfit(x,y,3)                 %用曲线拟合，计算 3 阶多项式系数
p =
  -0.1133   -0.0686   1.0238   -0.0011
s =                                      %有关误差估计的结构数组
    R:[4x4 double]                       % Vandermonde 矩阵的 Cholesky 因子，由正交分
                                           解[Q R] = qr(V,0)取得，V 为 x 的 Vander-
                                           monde 矩阵
    df:6                                 %自由度 = length(y) - (n + 1)
  normr: 0.0034                          %误差向量的 2 范数
≫y1 = polyval(p,x)'                     %多项式的计算值
y1 =
  -0.0011   0.1749   0.3431   0.4999   0.6417   0.7648   0.8657   0.9408   0.9863
0.9988
≫err = y' - y1                          %正弦函数与多项式值的误差向量
err =
  0.0011   -0.0013   -0.0011   0.0001   0.0011   0.0012   0.0003   -0.0011   -
0.0015   0.0012
≫norm(err)                              %误差 2 范数与 norm 相一致
ans =
  0.0034
```

【例 6-23】 今有一组试验数据 $\boldsymbol{x}$ = [0 1 2 3 4 5 6 7]，$\boldsymbol{y}$ = [0 0.64 0.87 0.93 0.96 0.98 0.99 0.995 1.01]，试用 4 阶多项式拟合。

解：

```
≫x = [0:7]';                            %输入向量 x
≫y = [0 0.64 0.87 0.94 0.96 0.98 0.99 1.01]'; %输入向量 y
≫table = [x,y]                          %x、y 列表
table =

         0         0
```

```
        1.0000      0.6400
        2.0000      0.8700
        3.0000      0.9400
        4.0000      0.9600
        5.0000      0.9800
        6.0000      0.9900
        7.0000      1.0100
>>[p s] = polyfit(x, y, 4)                          %计算 4 阶多项式拟合
p =
    -0.0024     0.0454     -0.3046     0.8851     0.0038
s =                                                 %结构数组 s
        R: [5x5 double]
         df: 3
    normr: 0.0256
>>y1 = polyval(p, x)                                %计算多项式的值
y1 =
    0.0038
    0.6273
    0.8800
    0.9466
    0.9537
    0.9694
    1.0031
    1.0061
>>plot(x, y, 'or', x, y1, '-k')                     %试验数据与拟合曲线如图 6-1 所示
>>grid on
```

【例 6-24】 某厂生产直流调速装置，电流规格见向量 $\boldsymbol{A}$ = [35 70 110 150 180 270 360 450]，单位安培。其相对应的价格，符合向量price = [1.49 1.54 1.72 1.78 2.05 2.4 3.35 3.8]，单位为万元，求 300A、700A 规格的直流调速装置的估计价格。

解：

对上述数据进行 2 阶多项式拟合，设系数多项式向量为 p

```
>>A = [35,70,110,150,180,270,360,450];              %输入规格向量 A
>>price = [1.49,1.54,1.72,1.78,2.05,2.4,3.35,3.8];  %输入价格向量
>>p = polyfit(A, price, 2)                          %调用多项式拟合函数，计算向量 p
p =
    0.0000     0.0023     1.3604
>>A1 = [300,700];                                   %输入待求规格向量 A1
>>pr = polyval(p, A1)                               %300A，700A，规格的价格估算值分
                                                      别为 2.7112 万元和 6.5965 万元
```

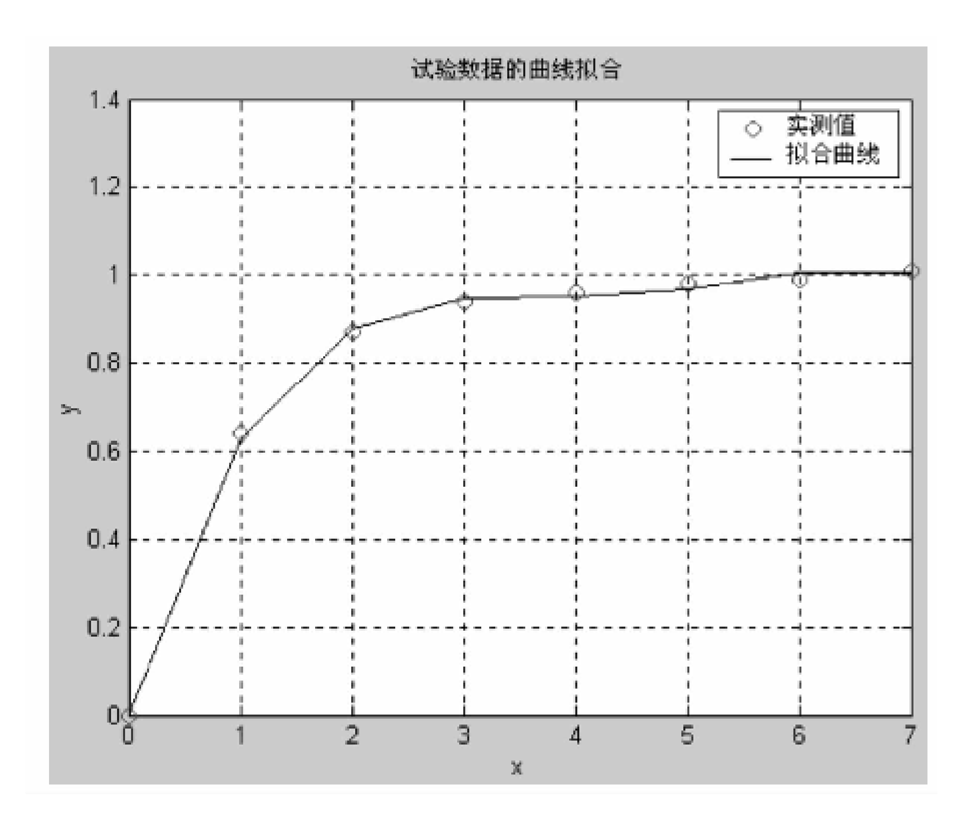


图 6-1 试验数据的多项式拟合

```
pr =
    2.7112    6.5965
```

6.11 一维插值

在生产和科学实验中，我们往往只掌握有限的测试数据，例如 $y = f(x)$，在区间 $[a, b]$ 上，测得的数据为 x_i，y_i，其中 $i = 1, 2, 3, \cdots, n$。对于在区间上的其他数据，只能进行估计。这种在已知数据中，用较简单的数学函数通过它，并对邻近数据进行估值计算称为插值。插值的方法在数学上有很多种，如线性插值、二次插值、样条插值和拉格朗日插值等。

6.11.1 拉格朗日多项式插值

拉格朗日多项式插值公式比较直观，它的正确性用插入点代入即可得到证明。为简化起见，设有 4 个测试点，测试数据由 $\boldsymbol{X} = [x_1\ x_2\ x_3\ x_4]$和 $\boldsymbol{Y} = [y_1\ y_2\ y_3\ y_4]$表示，则拉格朗日插值多项式的公式为

$$\begin{aligned} p(x) = &(x - x_2)(x - x_3)(x - x_4)y_1/(x_1 - x_2)/(x_1 - x_3)/(x_1 - x_4) + \\ &(x - x_1)(x - x_3)(x - x_4)y_2/(x_2 - x_1)/(x_2 - x_3)/(x_2 - x_4) + \\ &(x - x_1)(x - x_2)(x - x_4)y_3/(x_3 - x_1)/(x_3 - x_2)/(x_3 - x_4) + \\ &(x - x_1)(x - x_2)(x - x_3)y_4/(x_4 - x_1)/(x_4 - x_2)/(x_4 - x_3) \end{aligned} \tag{6-31}$$

式中，x 为新的插入点横坐标，显然这 4 阶多项式 $p(x)$ 通过测试点 (x_1, y_1)，(x_2, y_2)，(x_3, y_3)，(x_4, y_4)。由式（6-31）可以推广到 n 阶拉格朗日插值公式。

由式（6-31）可知，拉格朗日插值多项式的阶次比所通过的测试点的点数少 1。

【例 6-25】 今有正弦曲线 $y = \sin x$，已知 4 点在这曲线上，这 4 点用向量表示为

$\boldsymbol{X}=[0\ \ \pi/3\ \ 2\pi/3\ \ \pi]$，$\boldsymbol{Y}=[\sin(0)\ \ \sin(\pi/3)\ \ \sin(2\pi/3)\ \ \sin\pi]$，求拉格朗日插值多项式。

解：

由于Y(1) = Y(4) = 0，所以式（6-31）只剩下两项。将参数代入式（6-31），经化简后，通过 $\boldsymbol{X}$、$\boldsymbol{Y}$ 向量的拉格朗日多项式的程序表达式为

P(x) = 9* sqrt(3)/4/pi^2*（pi - x）.* x　　　　(6-32)

为了比较正弦曲线与插值曲线的差异，用下面语句绘制图形作比较，如图 6-2 所示。

```
>>x = 0:pi/50:pi;                        %设置 x 自变量
>>y1 = sin(x);                           %正弦波向量
>>y2 = 9* sqrt(3)/4/pi^2* (pi - x).* x;  %插值向量
>>plot(x,y1,'-k',x,y2,':r')              %绘制 y1，y2，如图 6-2 所示
```

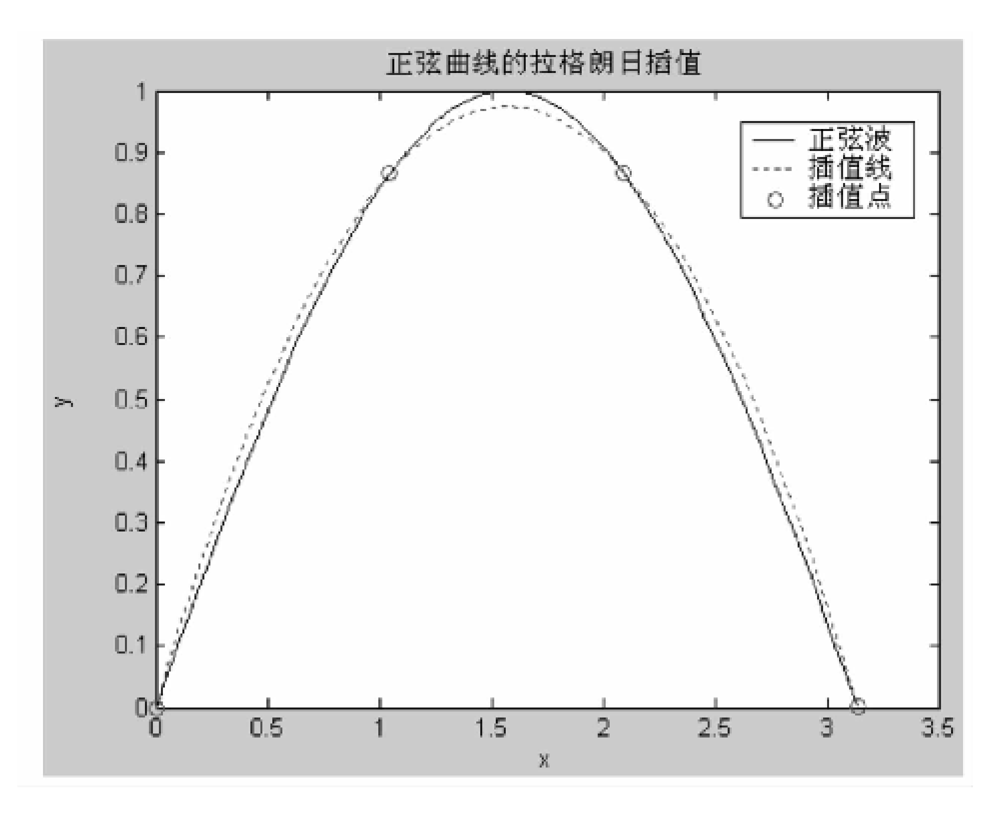


图 6-2　正弦曲线与拉格朗日插值曲线的对比

```
>>hold on                                %图形保持
>>X = [0,pi/3,2* pi/3,pi];               %4 点插值向量
>>Y = sin(X);
>>plot(X,Y,'or')                         %绘制插值点
>>err = abs(y1 - y2);                    %插值与正弦波的误差
>>max(err)                               %最大误差
ans =
    0.0434
```

用拉格朗日多项式插值公式，对于通过较多插入点的情况下，由于阶次较高，计算比较复杂。利用下面的计算程序可以使计算工作程序化。

```
%This program for calcuiate the Lagrange insert    %程序提示，X、Y为已知插入点向量，Xs为新的插值输入，Ys为相应于Xs的插值输出
%The X,Y are know vactor,Xs is insert datum of input
%and Ys is the datum of calculate by Lagrange method
function Ys = funlag(X,Y,Xs)                       %函数定义项
n = length(X);                                     %测试X的长度
n1 = length(Xs);                                   %测试Xs的长度
for i = 1:n1                                       %计算Ys的循环次数
    x = Xs(i);                                     %输入插值赋予x
    s = 0;                                         %多项式的代数和s，赋初值
    for j = 1:n                                    %计算s的循环
      p = 1;                                       %多项式乘积赋初值
      for k = 1:n                                  %计算多项式乘积的循环
        if k ~ = j                                 %若k、j不等则执行多项式乘积计算。否则滑过
          p = p* (x - X(k))/(X(j) - X(k));
        end
      end
    s = s + p* Y(j);
      end
      Ys(i) = s;                                   %插值输出
    end
```

【例 6-26】 已知 $\boldsymbol{Y}=\exp(-\boldsymbol{X})\sin(2\boldsymbol{X})$，当 $\boldsymbol{X}=[0\ 1\ 2\ 3\ 4\ 5\ 6]$时，$\boldsymbol{Y}=[00.3345-0.1024-0.0139\ 0.0181-0.0037-0.0013]$。

用拉格朗日插值，使插值多项式通过上述插入点，并绘制插值曲线。

解:

```
>>X = 0:6;                                         %输入X
Y = exp(-X).* sin(2* X)                            %计算Y
Y =
    0   0.3345   -0.1024   -0.0139   0.0181   -0.0037   -0.0013
plot(X,Y,'or')                                     %绘制插入点
hold on                                            %图形保持
Xs = 0:0.2:6;                                      %设置新的插值点
Ys = funlag(X,Y,Xs);                               % 调用Lagrange插值多项式程序，计算插值输出
plot(Xs,Ys,'-k')                                   %绘制插值曲线
Y1 = exp(-Xs).* sin(2* Xs);                        %计算原来的衰减正弦曲线
plot(Xs,Y1,':k')                                   % 绘制衰减正弦曲线与插值曲线相比较。如图6-3所示
```

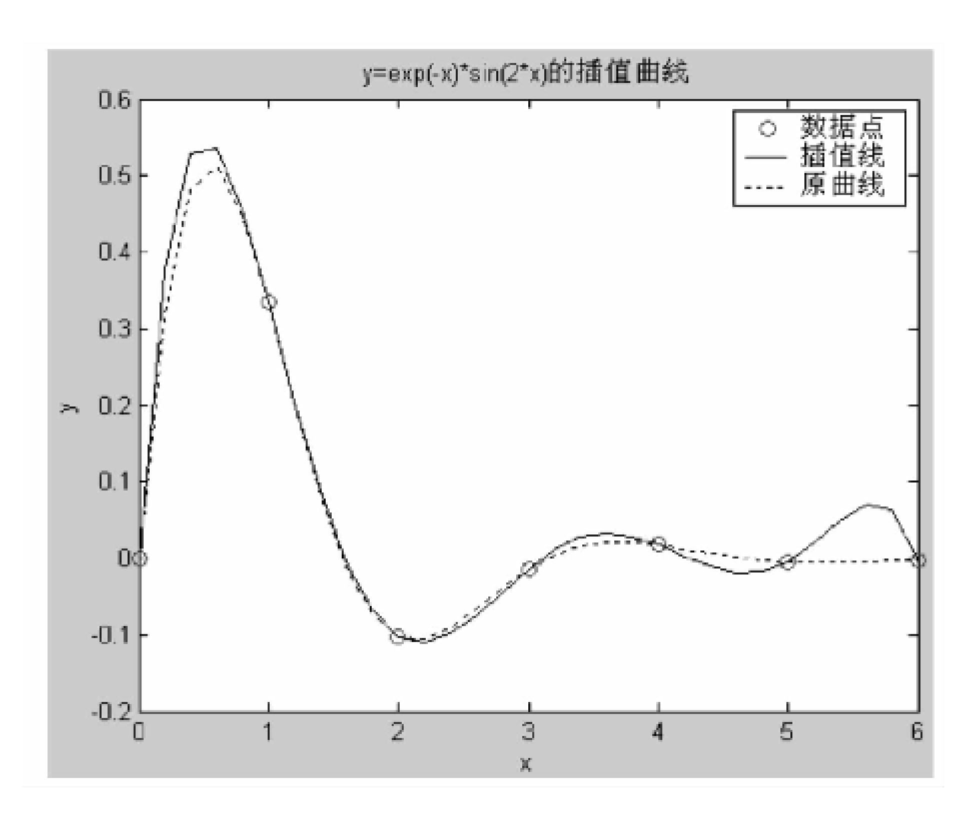


图 6-3　衰减正弦曲线与插值曲线

由图 6-3 可知，插值曲线在［0，4］区间内误差较小，在［4，6］区间则误差较大。

6.11.2　MATLAB 的一维插入函数

MATLAB 的一维插入函数的书写格式为

YI = interp1(X, Y, XI, 'method')

式中，X、Y 为已知插值点向量（即已知测试点向量），如图 6-3 上圆点所示。XI 为新的插入点输入向量，YI 为新的插入点输出向量。Method 指插入方法。对于插入方法有以下 4 种：

(1) 邻近插入（Nearest neighbor interpolation）用"nearest"写入插值公式中的 method 位置，这种插值的取值，是等于最邻近插入点的输出。这种插值，计算方便，但精度较差。

(2) 线性插入，（Linear interpolation（default），用"linear"写入插值公式中的 method 位置，这种插值是将已知插入点的输出用直线连接起来。这种插值也是缺省的插值法。

(3) 立方样条插值（Cubic spline interpolation）用"spline"写入插值公式中的 method 位置，这种插值在每段都用 3 次多项式表示，且 1、2 阶的导数连续。因此保证曲线的光滑。

(4) 立方插值（Piecewise cubic Hermite interpolation）用"cubic"写入插值公式中的 method 位置，它是采用分段的 Himete 插值方法。类似于 spline 插值。

【例 6-27】　已知衰减正弦曲线，$Y = \exp(-X)\sin(2X)$，$X = [0\ 0.5\ 1 1.5\ 2\ 2.5\ 3\ 3.5\ 4\ 4.5\ 5\ 5.5\ 6]$，求邻近插值和线性插值，并且用曲线显示。

解：

```
>>X = [0:0.5:6];                          %插值点输入向量
>>Y = exp(-X).*sin(2*X);                  %插值点输出向量
>>XI = [0:0.1:6];                         %新设置插值点输入向量
```

```
>>Ya = interp1(X,Y,XI,'nearest');          %邻近插值输出向量
>>Yl = interp1(X,Y,XI);                    %线性插值输出向量
>>plot(X,Y,'o',XI,Ya,'-',XI,Yl,':')        %绘制插值曲线图形
>>hold on                                  %图形保持
>>Ys = exp(-XI).*sin(2*XI);                %计算原始的衰减正弦曲线的纵坐标
>>plot (XI, Ys,'-.')                       %绘制原始的衰减正弦曲线，如图 6-4 所示
```

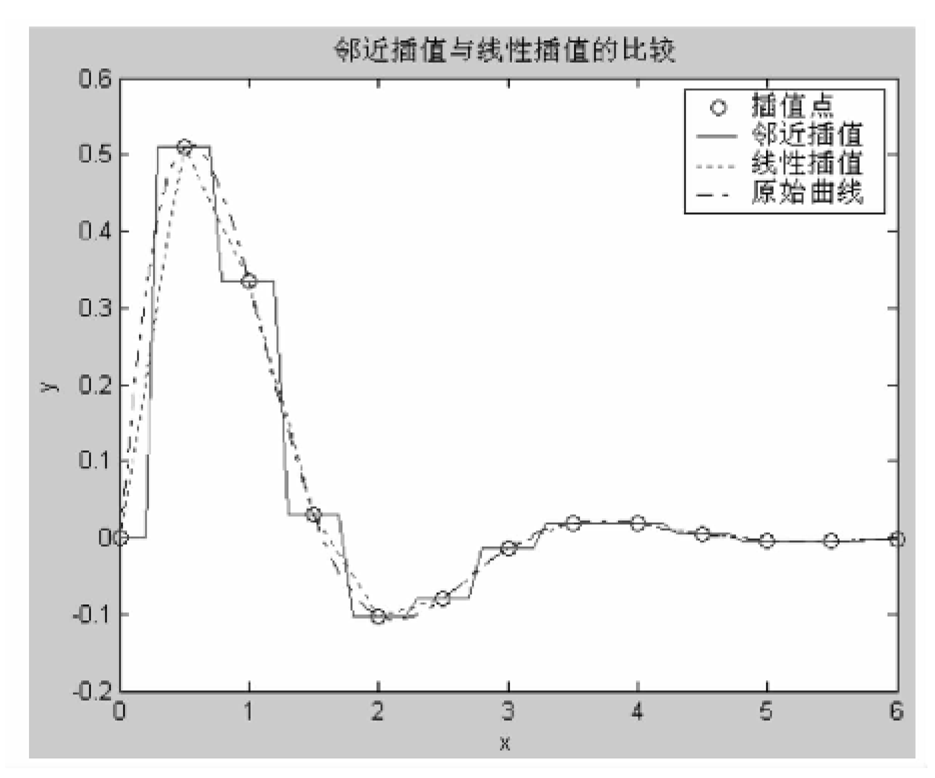


图 6-4　用邻近插值和线性插值绘制的衰减正弦曲线

【例 6-28】　某工厂生产变频器的容量等级，依照向量 $\boldsymbol{X}$ = [75 90 110 132 200 220]，单位为 kW。与之相对应的价格向量为 $\boldsymbol{Y}$ = [5.46 6.51 7.49 8.36 10.58 13.5]，单位为万元。今有用户订购 100kW 和 160kW 各 1 台，问如何估算单价。

解：

考虑采用线性插值法来估算单价。

```
>>X = [75,90,110,132,200,220];            %输入容量等级向量
>>Y = [5.46,6.51,7.49,8.36,10.58,13.5];   %输入价格向量
>>XI = [100,160];                         %设置容量插值向量
>>YI = interp1(X,Y,XI)                    %用线性插值求价格插值为 7 万元和 9.2741 万
                                            元。它对应容量为 100kW 和 160kW
YI =
    7.0000      9.2741
```

6.12　二维插值

二维插值主要用于二维函数的估值和图像处理。它的书写格式与一维插值相似。它的书写格式为

```
ZI = interp2(X,Y,Z,XI,YI)
ZI = interp2(Z,XI,YI)
ZI = interP2(Z,ntime)
ZI = interp2(X,Y,XI,YI,method)
```

式中，X、Y、Z 为样本数据矩阵，XI、YI、ZI 为插值数据矩阵，ntime 为在元素间插入扩充数据的回归次数。method 为选取插值的方法。插值方法有以下几种：

(1) 邻近插值，method = nearest

(2) 双线性插值 method = linear

(3) 样条插值 method = spline

(4) 立方插值 method = cubic

【例 6-29】 已知三维函数为 $Z = X\exp(-X^2 - Y^2)$，样本数据[X,Y] = meshgrid(-3:3)，插值数据为[XI,YI] = meshgrid(-3:0.25:3)，求样本数据表面图、插值方法为邻近插值、线性插值、样条插值和立方插值情况下的表面图。

解：

```
≫[X,Y] = meshgrid(-3:3)                      %生成网格坐标
X =

    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3
    -3    -2    -1     0     1     2     3

Y =

    -3    -3    -3    -3    -3    -3    -3
    -2    -2    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1    -1    -1
     0     0     0     0     0     0     0
     1     1     1     1     1     1     1
     2     2     2     2     2     2     2
     3     3     3     3     3     3     3
≫Z = X.*exp(-X.^2-Y.^2);                     %计算在网格坐标下的纵坐标矩阵
≫[Xi,Yi] = meshgrid(-3:0.25:3);              %生成插值情况下的网格坐标
≫surf(X,Y,Z)                                 %绘制松散坐标下的样本表面图，如图 6-5 所示
≫Zin = interp2(X,Y,Z,Xi,Yi,'nearest');       %计算二维邻近插值
≫figure                                      %建立图形窗口
≫surf(Xi,Yi,Zin)                             %绘制邻近插入表面图如图 6-6 所示
≫figure                                      %建立图形窗口
```

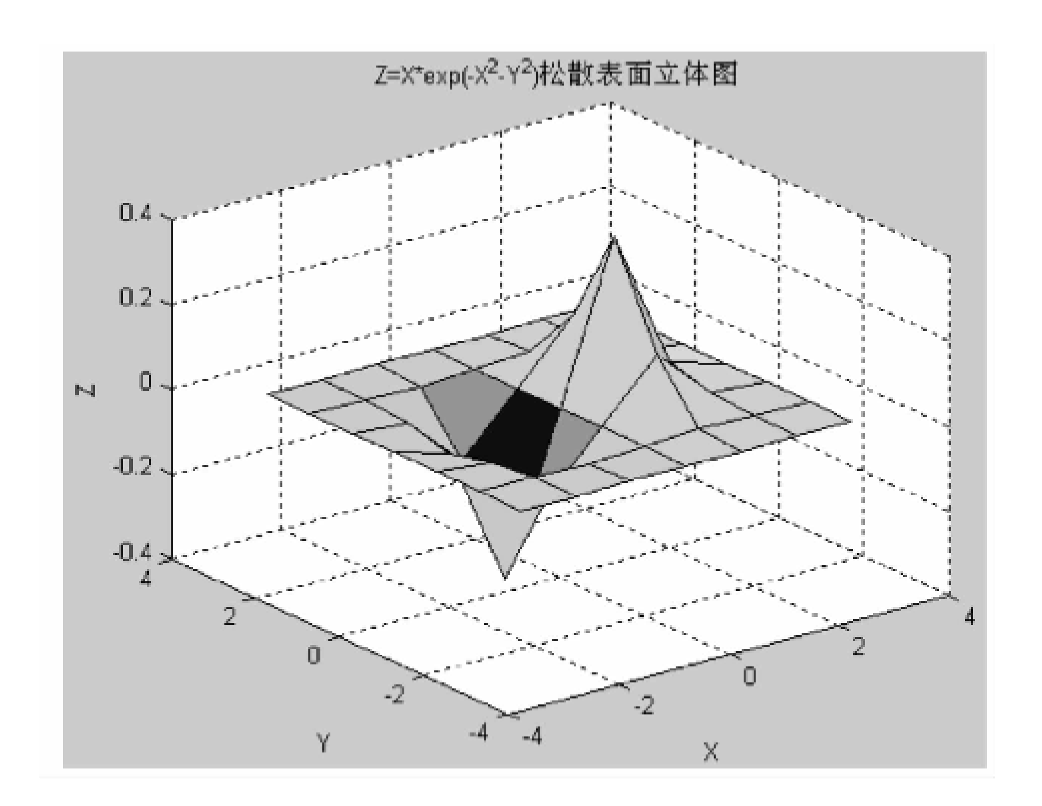


图 6-5　松散的样本表面图

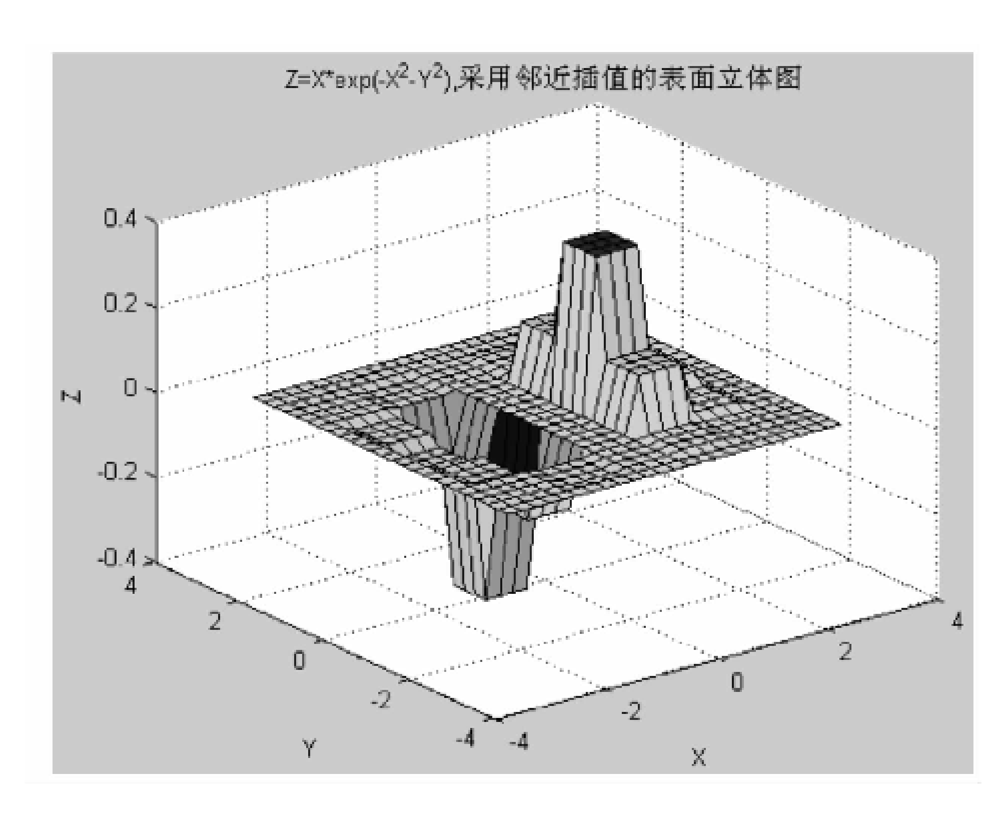


图 6-6　邻近插值表面图

```
≫Zil = interp2(X,Y,Z,Xi,Yi,'linear');        %计算线性插值
≫surf(Xi,Yi,Zil)                              %绘制线性插值表面图如图 6-7 所示
≫figure                                       %建立图形窗口
```

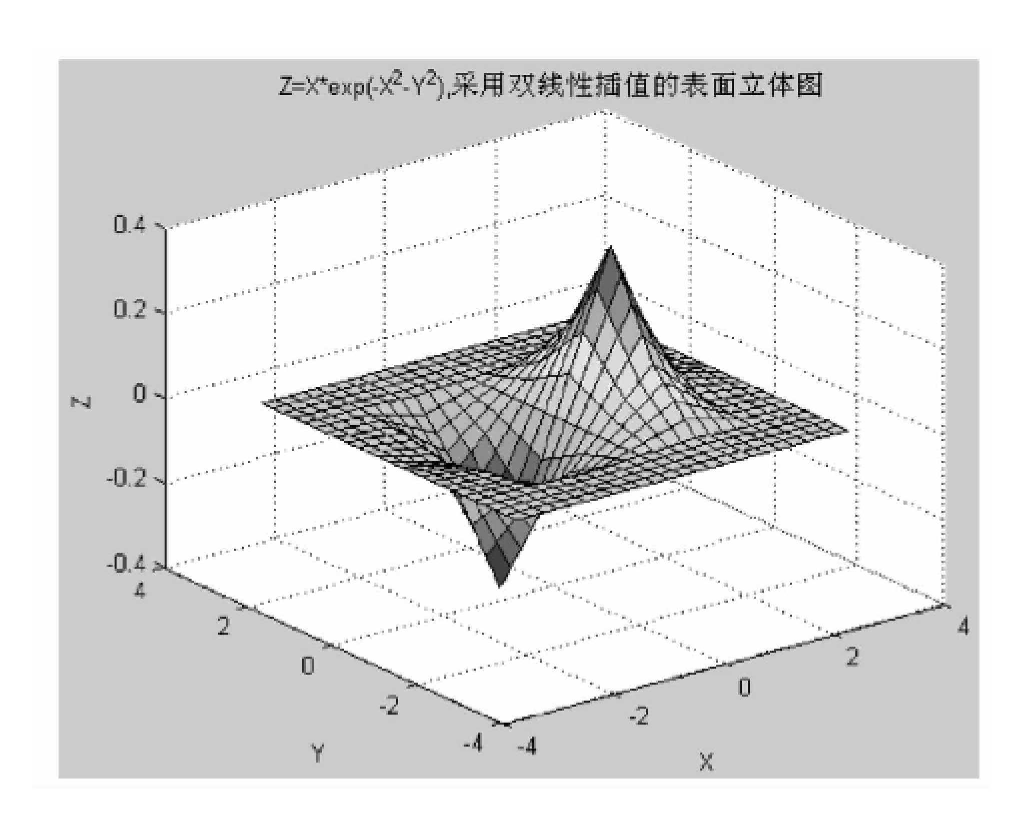


图 6-7　线性插值表面图

```
≫Zic = interp2(X, Y, Z, Xi, Yi, 'cubic');          %计算立方插值
≫surf(Xi, Yi, Zic)                                  %绘制立方插值表面图如图 6-8 所示
```

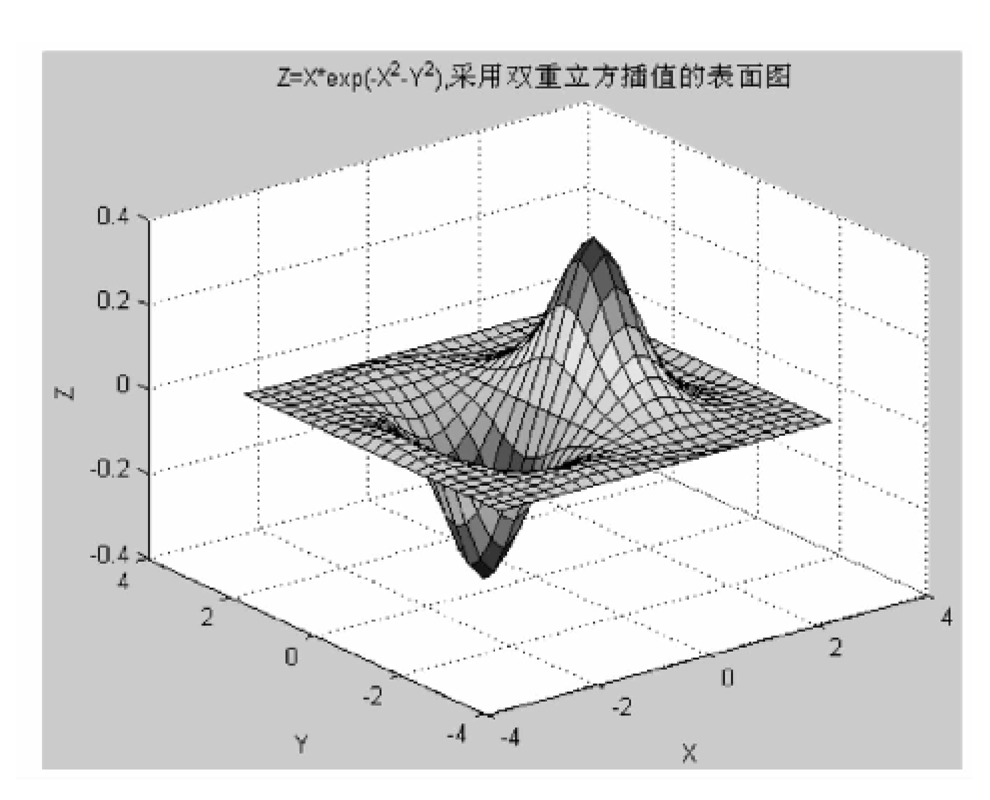


图 6-8　立方插值表面图

【例 6-30】 已知矩阵 $A=\begin{pmatrix}8&8&8&8&8&8&8&8\\8&4&4&4&4&4&4&8\\8&4&2&2&2&2&4&8\\8&4&2&1&1&2&4&8\\8&4&2&1&1&2&4&8\\8&4&2&2&2&2&4&8\\8&4&4&4&4&4&4&8\\8&8&8&8&8&8&8&8\end{pmatrix}$，绘制表面图，经过 ntime = 2，回归插值后再绘制表面图。

解：

```
>>A = 8* ones(8);                              %输入元素为 8 的 8 阶方阵
>>A1 = 4* ones(6); A(2:7,2:7) = A1;            %输入内 1 环为 4 的 6 阶方阵
>>A1 = 2* ones(4); A(3:6,3:6) = A1;            %输入内 2 环为 2 的 4 阶方阵
>>A1 = ones(2); A(4:5,4:5) = A1                %输入内 3 环为 1 的 2 阶方阵
A =
     8     8     8     8     8     8     8     8
     8     4     4     4     4     4     4     8
     8     4     2     2     2     2     4     8
     8     4     2     1     1     2     4     8
     8     4     2     1     1     2     4     8
     8     4     2     2     2     2     4     8
     8     4     4     4     4     4     4     8
     8     8     8     8     8     8     8     8
>>surf(A)                                      %绘制表面图，如图 6-9 所示
```

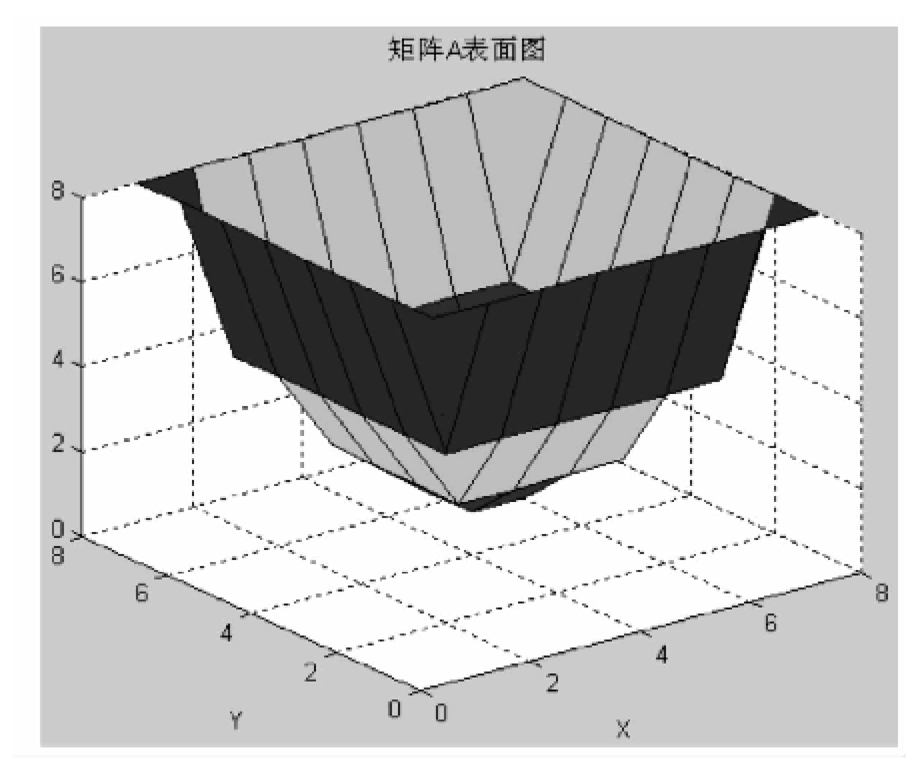


图 6-9 矩阵 A 表面图

```
>>figure                          %建立图形窗口
>>ZI = interp2(A,2);              %取 ntime = 2，计算扩展的回归插值
>>surf(ZI)                        %绘制表面图如图 6-10 所示
```

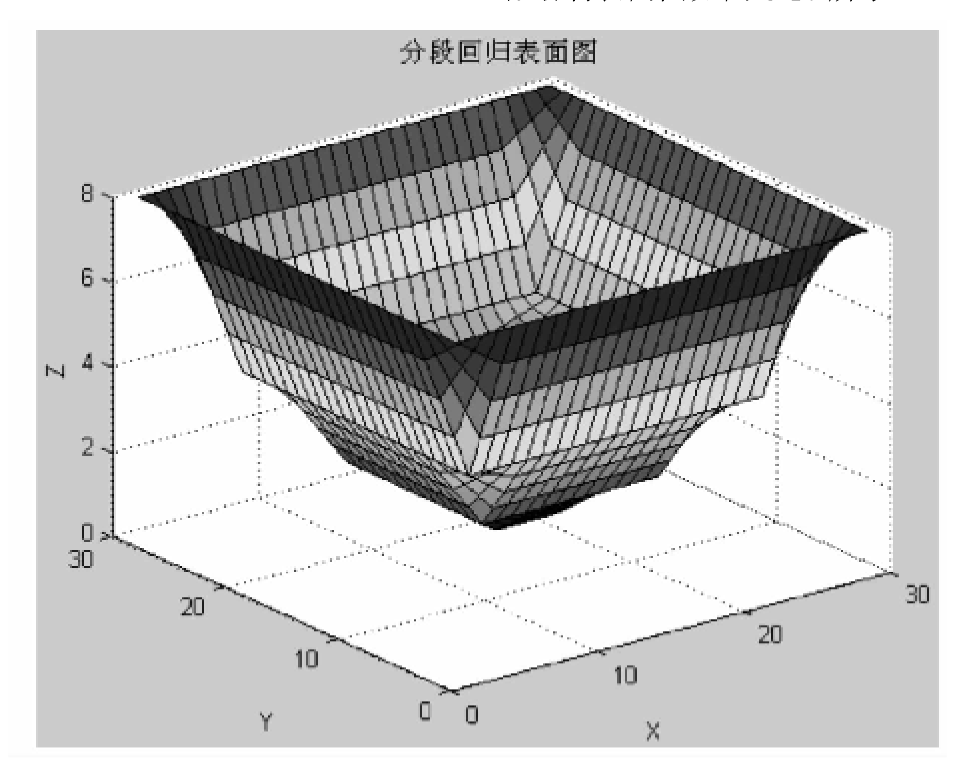


图 6-10　分段回归表面图

第 6 章习题

6-1　用求根函数 roots 计算下列多项式的根：

(1) $x^3 + 3x^2 + 5x + 7 = 0$

(2) $x^5 + 14x^4 + 84x^3 + 244x^2 + 323x + 150 = 0$

6-2　已知多项式的根为 1.5、2.5、2 + 2i、2 − 2i，求此多项式。

6-3　已知多项式 $p_1 = x^3 + 3x + 1$；$p_2 = x^5 + 3x^4 + 5x^3 + 8x^2 + 13x + 5$

　　求多项式的乘积 p_1p_2。

6-4　已知多项式系数 **p**1 = [1 2 1]；**p**2 = [1 5 7 13 21 1]，求 **p**2/**p**1。

6-5　已知多项式 $p(x) = x^5 + 7x^3 + 4x^2 + 1$，求 **x** = [1 2 3 4 5 6 7 8 9 10]时，求多项式的值。

6-6　已知多项式 $y = x^5 + 7x^3 + 4x^2 + 1$，求其一阶微商。

6-7　已知多项式 $y = x^3 - 5x^2 - 11x + 5$，求其不定积分。

6-8　已知分式多项式

$$y = \frac{x+3}{x^2+7} + \frac{x+8}{x^3+3x+1},$$

求通分后的分子多项式和分母多项式。

6-9　已知分式多项式

$$y = \frac{x^3+3x+1}{(x+1)(x+3)(x+5)(x+7)},$$

将其分解成部分分式。

6-10　已知测试数据如下（见表 6-3）

表 6-3 某测试数据

x	1	2	3	4	5	6	7	8	9	10	11	12
y	6.4	7.4	11.6	14.4	15.8	21.3	24.6	27	31.2	35.1	39.1	44.6

求二次曲线拟合的多项式系数。

6-11 已知 $y = \sin^2 x / (1 + x^2)$，试将其展开成 7 级 Taylor 级数。

6-12 已知 $y = \sin^2 x$，当 $\boldsymbol{x}_1$ = [0 pi/4 pi/2 3pi/4 pi]时 $\boldsymbol{y}_1$ = [0 1/2 1 1/2 0]，试用线性插值计算 x2 = [0:pi/18:pi]时的 y2，并绘制误差曲线 y2 - y。

第 7 章　MATLAB 在初等数学中的应用

MATLAB不但为科学计算提供了强有力的工具，同时也为初等数学的演算实现自动化。它能实现数学表达式的化简，数学表达式的展开、合并，级数求和，求数组的最大公约数、最小公倍数，列写素数表，线性方程组的整数解，数列的排序和初等几何命题的证明等。在本章中引用的 MATLAB 函数的名称和功能见表 7-1。

表 7-1　MATLAB 用于初等数学的函数名称和功能

函　数　名	用　法　及　功　能
primes	素数列表，primes (n)返回小于等于 n 的所有素数组成的行向量
isprime	用逻辑判断是否为素数，如为素数，则返回 1，否则为 0
factor	分解为素数因子的乘积，亦用于符号表达式的因式分解
prod	计算数组元素沿不同维的乘积
expand	将符号表达式进行展开，适用于多项式、三角函数、指数函数和对数函数
collect	将符号表达式进行同类项合并
simple	利用 factor、expand、collect 将符号表达式进行简化，使其长度为最短
simplify	简化符号矩阵的每个元素
mod	模数，除后带符号的余数，mod(x,y) = x-y.* floor(x./y)
rem	除后的余数，rem(x,y) = x-y.* fix(x./y)
sum	向量元素之和，以数值表示。对矩阵为列向量之和以行向量表示
symsum	符号变量级数求和
isequal	检验数组是否相等
numel	计算向量或矩阵所包含的元素数
numden	提取分式多项式的分子及分母
sort	对向量元素按升序排列，对矩阵则按列进行升序排列
sortrows	按矩阵行首的大小以升序排列
syms	设置符号变量
subs	变量替换，用新的变量替代旧的变量
fix	沿零取整
floor	沿 - inf 取整
round	就近取整
ceil	沿 + inf 取整
solve	解方程组
reshape	改变矩阵的大小
polyarea	计算多边形面积

下面分别介绍 MATLAB 在初等数学中的应用。

7.1 素数的计算

某一个数，除了 1 和它本身以外，没有其他因子，这个数称为素数。例如 2、3、5、7、101、103、107、109 等均为素数。因为它除了 1 和它本身以外，没有数能除尽它。MATLAB 提供计算素数表的函数 primes，它的书写格式如下：

primes(n)

它返回一个小于或等于 n 的素数列表。检测某整数 k 是否是素数的方法是用奇数 3、5、7、9…直到小于或等于$\sqrt{k}$的整数去试除 k，若都不能除尽，则 k 为素数，否则 k 不是素数(证明从略)。对于素数列表，则可以从奇数列表中，逐个删除能被奇数 3、5、7、9…直到小于或等于$\sqrt{n}$整除的数，剩下的则全是素数，素数列表即可完成。Primes (n)的程序如下：

```
function p = primes(n)                                          %函数定义行
%PRIMES Generate list of prime numbers.                         %函数标题行
%PRIMES(N) is a row vector of the prime numbers less than or    %help 内容
%equal to N. A prime number is one that has no factors other
%than 1 and itself.
%
%See also FACTOR，ISPRIME.
%Copyright 1984-2001 The MathWorks，Inc.
%$Revision: 1.15 $   $Date: 2001/04/15 12:01:45 $
                                                                %以下为函数体
if length(n) ~ = 1，error('N must be a scalar')；end            % 假若 n 不是标量，则出错
if n < 2，p = zeros(1,0)；return，end                          % 假若 n 小于 2，则显示 0，并返回
p = 1:2:n;                                                      % 建立奇数列表，即向量 p
q = length(p)；                                                 %计算向量 p 的长度
p(1) = 2;                                                       %赋予第一个素数为 2
for k = 3:2:sqrt(n)                                             % 以 3 以上的奇数作循环，直到 sqrt(n)
   if p((k + 1)/2)                                              % 若 p((k + 1)/2)存在，则执行下一语句
      p(((k * k + 1)/2):k:q) = 0;                               % 在数列 p 中，将有奇数 k 因子的项置零
   end                                                          %结束条件判断
end                                                             %执行 for 循环
p = p(p > 0)；                                                  % 保留大于零的素数项，素数列表完成
```

下面举例予以说明：

【例 7-1】 建立2～1000以内的素数表。

解:

```
≫prim = primes(1000);            % 找出 2～1000 内的素数向量,primes 是求素数向
                                   量函数,prim 为设定的素数表变量名
≫n = numel(prim)                 % 求素数向量 prim 的元素数,numel 为求向量或矩阵的元
                                   素数的函数

n =
  168
≫primtab = reshape(prim,21,8)    % 将向量 prim 重写成 21 行 8 列,式中 reshape 为改变向量
                                   或矩阵大小函数,primtab 为素数表名
   primtab =                       % 列出 2～1000 以内的素数表
     2    79   191   311   439   577   709   857
     3    83   193   313   443   587   719   859
     5    89   197   317   449   593   727   863
     7    97   199   331   457   599   733   877
    11   101   211   337   461   601   739   881
    13   103   223   347   463   607   743   883
    17   107   227   349   467   613   751   887
    19   109   229   353   479   617   757   907
    23   113   233   359   487   619   761   911
    29   127   239   367   491   631   769   919
    31   131   241   373   499   641   773   929
    37   137   251   379   503   643   787   937
    41   139   257   383   509   647   797   941
    43   149   263   389   521   653   809   947
    47   151   269   397   523   659   811   953
    53   157   271   401   541   661   821   967
    59   163   277   409   547   673   823   971
    61   167   281   419   557   677   827   977
    67   173   283   421   563   683   829   983
    71   179   293   431   569   691   839   991
    73   181   307   433   571   701   853   997
```

【例 7-2】 检查数列 317、503、809、1011、10097、100101，其中哪些是素数，哪些不是。

解:

```
≫v = [317,503,809,1011,10097,100101];   % 已知数列,写成向量 v
≫isprime(v)                             % 判别向量 v 的元素是否是素数,isprime 是素
                                          数检查函数,它返回一个同样大小的向量或矩
                                          阵,如果返回的对应元素为 1,则原数为素数;
```

```
                                         如果为 0,则原数不是素数
ans =                                    %默认变量显示,向量 v 的前 3 个数字为素数,
                                         后 3 个数字则不是。
     1     1     1     0     0     0
```

【例 7-3】 求1000～1100间有几个素数，并列出素数。

解:

```
≫A = primes(1100);                       %列出 1100 以下的素数
≫find(A > 1000)                          %寻找大于 1000 向量 A 元素的下标
ans =
  Columns 1 through 13
    169   170   171   172   173   174   175   176   177   178   179   180   181
  Columns 14 through 16
    182   183   184
≫n = 16;
≫A(169:184)
ans =                                    %核对
  Columns 1 through 6
    1009      1013      1019      1021      1031      1033
  Columns 7 through 12
    1039      1049      1051      1061      1063      1069
  Columns 13 through 16
    1087      1091      1093      1097
≫isprime(ans)
ans =
  Columns 1 through 13
    1     1     1     1     1     1     1     1     1     1     1     1     1
  Columns 14 through 16
    1     1     1
```

7.2 分解质因子

将一个正数数 n 分解为素数（即质因子）的乘积，而各个质因子是依升序排列，这称为分解质因子。分解质因子函数的书写格式为

factor(n)

下面举例予以说明。

【例 7-4】 将 $n = 12321$ 分解为质因子的乘积。

解:

```
≫factor(12321)          %将 12321 分解成质因子的乘积
ans =
    3     3     37     37
```

```
≫prod(ans)            %计算向量 ans 中各元素的积,以核对分解的正确性。prod 为求向量
                        或矩阵元素之积的函数,详见下一节
ans =
    12321
```

【例 7-5】 将 $n = 20790$ 分解为质因子。

解:

```
≫factor(20790)        %将 20790 分解成质因子的乘积
ans =
    2     3     3     3     5     7     11
≫prod(ans)            %核对分解的正确性
  ans =
      20790
```

7.3 数组的元素乘积

计算数组各元素沿着不同维的乘积的函数，MATLAB 的书写格式为

B = prod(A)

B = prod(A,dim)

若 A 为向量，则它返回各元素的乘积。

若 A 为矩阵，则它返回一个行向量，行向量的每一个元素，对应矩阵每一列作为向量元素的乘积。

若 A 为多维数组，则在第一维中执行各元素的乘积。

对于矩阵 A，若 dim = 1，则矩阵B = prod(A,dim)对应每列元素的乘积，对于 dim = 2，则矩阵 B 对应每行元素的乘积。现举例如下：

【例 7-6】 已知向量 $\boldsymbol{A}$ = [3 5 7 11 13]，求各元素的积。

解:

```
≫A = [3 5 7 11 13];   %已知向量 A
≫prod(A)              %向量 A 元素的乘积
ans =
     15015
≫factor(ans)          %分解标量 ans 为质因子
ans  =
    3     5     7     11     13
```

山此可见，在向量元素是素数时，prod 与 factor 是互为可逆的函数。

【例 7-7】 已知矩阵 A 为 4 阶 pascal 矩阵，求各列元素的积。

解:

```
  ≫A = pascal(4)            %已知矩阵 A
A =
     1     1     1     1
```

```
     1     2     3     4
     1     3     6    10
     1     4    10    20
>>prod(A)                     % prod(A)返回一个向量
ans =
     1    24   180   800
```

【例 7-8】 已知三维数组的 3 页为 A(:,:,1) = ones(3); A(:,:,2) = 2* ones(3); A(:,:,3) = 3* ones(3)，求此三维数组的积。

解：

```
>>A(:,:,1) = ones(3);           %矩阵的第 1 页
>>A(:,:,2) = 2* ones(3);        %矩阵的第 2 页
>>A(:,:,3) = 3* ones(3);        %矩阵的第 3 页
>>A                             %显示三维数组 A
A(:,:,1) =
     1     1     1
     1     1     1
     1     1     1
A(:,:,2) =
     2     2     2
     2     2     2
     2     2     2
A(:,:,3) =
     3     3     3
     3     3     3
     3     3     3
>>prod(A)                       %三维数组 A 的元素乘积
ans(:,:,1) =                    %第一页各列之积
     1     1     1
ans(:,:,2) =                    %第二页各列之积
     8     8     8
ans(:,:,3) =                    %第三页各列之积
    27    27    27
```

【例 7-9】 已知矩阵 ***A*** 为 3 阶魔方阵，求 prod(A,1)、prod(A,2)和矩阵 ***A*** 所有元素的乘积。

解：

```
>>A = magic(3)               %已知矩阵 A
A =
     8     1     6
     3     5     7
```

```
    4    9    2
>>prod(A,1)              %计算列元素的乘积,以行向量表示
ans =
    96    45    84
>>prod(A,2)              %计算行元素的乘积,以列向量表示
ans  =
    48
   105
    72
>>prod(prod(A))          %2次求矩阵元素的积,即为矩阵所有元素积,因为第1次为矩阵
                           A列元素之积,构成的行向量,第2次为行元素之积,构成矩阵A
                           所有元素之积
ans =
      362880
>>factorial(9)           %核对计算矩阵A元素之积的正确性。factorial(n)为MATLAB
                           中求阶乘的函数,其中n为正整数,factorial(n)即数学表示式中
                           的n!,但MATLAB不认识n!,若在命令窗口输入n!,会显示出
                           错信息
ans =
      362880
```

7.4 数组元素之和

在MATLAB中求向量V元素之和的函数为

sum(V)

对于矩阵A，则sum(A)返回1个行向量，向量中每一个元素代表矩阵A中每一列元素之和。

格式　sum(A,1)

则与sum(A)相同。

格式　sum(A,2)

则返回一个列向量，它的每一列元素代表矩阵A相应行元素之和。现举例如下：

【例7-10】　已知向量$\boldsymbol{V}=[1\ 2\ 3\ \sqrt{2}\ 3/4\ 10^2]$，求元素之和。

解：

```
>>V = [1 2 3 sqrt(2) 3/4 10^2]   %已知向量V
V =
    1.0000   2.0000   3.0000   1.4142   0.7500   100.0000
>>sum(V)                         %向量元素之和
ans =
    108.1642
```

【例 7-11】 已知均匀分布随机矩阵 ***A*** 为 3 行 4 列，求矩阵元素之和。

解:

```
≫A = rand(3,4)                   %已知 3 行 4 列的均匀分布随机矩阵 A
A =
    0.9501   0.4860   0.4565   0.4447
    0.2311   0.8913   0.0185   0.6154
    0.6068   0.7621   0.8214   0.7919
≫sum(A)                          %矩阵 A 列元素之和
ans =
    1.7881   2.1394   1.2964   1.8521
≫sum(ans)                        %矩阵 A 所有元素之和
ans =
    7.0759
```

【例 7-12】 以分数格式计算 5 阶 Hilbert 矩阵的主对角线元素和副对角线元素之和。

解:

```
≫format rat                      %设置分数显示格式
≫A = hilb(5)                     %5 阶 Hilbert 矩阵
A =
    1       1/2     1/3     1/4     1/5
    1/2     1/3     1/4     1/5     1/6
    1/3     1/4     1/5     1/6     1/7
    1/4     1/5     1/6     1/7     1/8
    1/5     1/6     1/7     1/8     1/9
≫sum(diag(A))                    %主对角线元素之和
ans =
    563/315
≫sum(diag(fliplr(A)))            % 将矩阵 A 左右对换(采用 fliplr 函数),再求对角线元素
                                   之和,即为副对角线元素之和
ans =
      1
≫trace(A)                        %也可以用 trace 求主对角线元素之和
ans =
    563/315
≫trace(fliplr(A))
ans =
      1
```

7.5 数组元素的累加和

数组元素的累加和书写格式如下:

cumsum(A)

cumsum(A,dim)

若 A 为向量，则它返回一个向量，它的元素是相应于向量 A 的元素累加和。

若 A 为矩阵，则它返回一个同样大小的矩阵，它的元素是相应于矩阵 A 的列元素的累加和。

dim 指维数，若 dim = 1，则 cumsum (A,dim)与 cumsum(A)相一致。若 dim = 2，则它的元素是对应于矩阵 A 的行元素的累加和。现举例予以说明。

【例 7-13】 已知向量 $\boldsymbol{A}$ = [1 2 3 4 5]，求 cumsum(A)。

解:

```
≫A = [1 2 3 4 5]            %已知向量 A
A =
     1     2     3     4     5
≫cumsum(A)                  %向量 A 的累加和。累加和向量的第 1 项等于向量 A 的第 1
                              项,累加和的第 2 项等于向量 A 的 1、2 项之和,累加和的第 3
                              项等于向量 A 的第 1、2、3 项之和,依此类推
ans =
     1     3     6    10    15
```

【例 7-14】 已知矩阵 $\boldsymbol{A}$ 为 4 行 5 列的单位矩阵，求cumsum(A,1)及 cumsum(A,2)。

解:

```
≫A = ones(4,5)              %已知 4 行 5 列的单位矩阵
A =
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
     1     1     1     1     1
≫cumsum(A,1)                %沿列执行矩阵的累加和
ans =
     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4
≫cumsum(A,2)                %沿行执行矩阵的累加和
ans =
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
```

【例 7-15】 使用数组累加函数，编制任意阶的 Pascal 矩阵。

解:

程序如下：

```
%This program for creat matrix of Pascal
function y = pasl(n)              %函数定义项，函数名为 pasl(为了区别于 MATLAB
                                    的 Pascal 矩阵函数)
A = ones(1,n);                    %设置 Pasl 矩阵的首行，首列
B = zeros(n);                     %确定矩阵的大小
B(1,:) = A;                       %为 Pasl 矩阵的首行赋值
for i = 2:n                       %设置 2~n 循环
    B(i,:) = cumsum(B(i-1,:));    %Pasl 矩阵的下一行为上一行的累加和
end
ans = B                           %用默认变量显示 Pasl 矩阵
```

程序执行举例如下：

```
≫pasl(4)
ans =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20

≫pascal(4)                        %与 pasl(4)相对照，完全一致
ans =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20

≫pasl(7)
ans  =

     1     1     1     1     1     1     1
     1     2     3     4     5     6     7
     1     3     6    10    15    21    28
     1     4    10    20    35    56    84
     1     5    15    35    70   126   210
     1     6    21    56   126   252   462
     1     7    28    84   210   462   924

≫pascal(7)                        %与 pasl(7)相对照，完全一致
ans =

     1     1     1     1     1     1     1
     1     2     3     4     5     6     7
```

```
     1     3     6    10    15    21    28
     1     4    10    20    35    56    84
     1     5    15    35    70   126   210
     1     6    21    56   126   252   462
     1     7    28    84   210   462   924
```

但 pasl 矩阵函数的程序比下列 MATLAB 的 pascal 矩阵函数的程序大为简洁。

```
>>type pascal
function P = pascal(n,k)
%PASCAL Pascal matrix.
%PASCAL(N) is the Pascal matrix of order N: a symmetric positive
%definite matrix with integer entries, made up from Pascal's
%triangle. Its inverse has integer entries.
%
%PASCAL(N,1) is the lower triangular Cholesky factor (up to signs
%of columns) of the Pascal matrix. It is involutary (is its own
%inverse).
%
%PASCAL(N,2) is a rotated version of PASCAL(N,1), with sign flipped
%if N is even, which is a cube root of the identity.

%Reference:
%[1] N. J. Higham, Accuracy and Stability of Numerical Algorithms,
%     Society for Industrial and Applied Mathematics, Philadelphia,
%     1996, Sec. 26.4.
%
%Nicholas J. Higham, Dec 1999.
%Author: N.J. Higham 6-23-89
%Copyright 1984-2001 The MathWorks, Inc.
%$Revision: 5.12 $ $Date: 2001/04/15 12:02:28 $

if nargin < 2, k = 0; end
P = diag((-1).^(0:n-1));
P(:,1) = ones(n,1);

% Generate the Pascal Cholesky factor (up to signs).
for j = 2:n-1
    for i = j+1:n
        P(i,j) = P(i-1,j) - P(i-1,j-1);
    end
```

```
end

if k == 0
    P = P* P';
elseif k == 1
    % P = P
elseif k == 2
    P = rot90(P,3);
    if n/2 == round(n/2), P = - P; end
else
    error('Second argument must be 0, 1, or 2.')
end
```

7.6 最大公约数 gcd

两个整数，除了 1 以外，都能被某个整数除尽，则这个整数被称之为公约数。在诸多公约数中必有一个最大的，称之为最大公约数。最大公约数在 MATLAB 中的表示格式如下：

```
G = gcd(A,B)
[G,C,D] = gcd(A,B)
```

式中，A、B 为整数数组，gcd 为最大公约数的函数名。G 返回一个最大公约数的数组，这个数组的每一个元素，对应 A、B 对应元素的最大公约数。数组 C、D 满足下列方程式：

```
A.* C + B.* D = G
```

这个方程式对于解 Diophantine（丢番图）方程式和 Hermite 变换的计算元素是有用的。下面举例说明。

【例 7-16】 已知向量 ***A*** = [16 25 54 49]、***B*** = [48 35 78 84]，求向量 ***A***、***B*** 的最大公约数向量。

解：

```
>>A = [16,25,54,49];
>>B = [44 35 78 84];
>>G = gcd(A,B)          %G 的第一列元素对应于 16、44 的最大公约数,G 的第二列元素对应
                          于 25、35 的最大公约数,G 的第三列元素对应于 54、78 的最大公约
                          数,G 的第四列元素对应于 49、84 的最大公约数
G =
    4     5     6     7
```

【例 7-17】 已知矩阵 ***A*** 为 4 阶 pascal 矩阵，矩阵 ***B*** 为 4 阶魔方阵，求最大公约数矩阵 ***G*** 和矩阵 ***C***、***D***。

解：

```
>>A = pascal(4)                    %已知 pascal(4)矩阵
```

```
A =

     1     1     1     1
     1     2     3     4
     1     3     6    10
     1     4    10    20
>>B = magic(4)                    %已知 magic(4)
B =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
>>[G C D] = gcd(A,B)
G =                               %最大公约数矩阵 G
     1     1     1     1
     1     1     1     4
     1     1     6     2
     1     2     5     1
C =

     1     1     1     1
     1    -5    -3     1
     1    -2     0    -1
     1    -3    -1     0
D =
     0     0     0     0
     0     1     1     0
     0     1     1     1
     0     1     1     1
>>A.*C + B.*D                     %核对 A.*C + B.*D = G
ans =
     1     1     1     1
     1     1     1     4
     1     1     6     2
     1     2     5     1
```

【例 7-18】 已知丢番图方程式 $30x+56y=8$，求 x、y。

解：

```
>>[g c d] = gcd(30,56)            %求标量 30、56 的最大公约数
g =
```

```
    2
c =
   -13
d =
    7
```

按定义，对于标量 c、d 应满足$30\times(-13)+56\times(7)=2$，乘倍数 8/2 = 4，则得 $30\times(-13\times4)+56\times(7\times4)=8$，比较这个式子与原始方程，可以观察到 $x=-13\times4=-52; y=7\times4=28$，这就是 $30x+56y=8$ 的解。

以上讨论的是两个数组的最大公约数，对于多个数值或多个数组的最大公约数，MATLAB 没有内装函数可利用，但可通过编制程序来解决。现以求多个数值最大公约数为例，在 M 文件编辑器中输入如下程序，程序名即为函数名 gcdv.m。将此程序存入当前工作目录，即可在程序窗口中调用此函数运行，求解向量中元素的最大公约数。

```
%本程序计算向量内元素的最大公约数      %程序注解
function gv = gcdv(V)                  %函数定义项,gcdv 为函数名,V 为输入向量
[m,n] = size(V);                       %测试向量 V 的大小
if m ~ = 1                             %若行数不是 1,则显示出错,指出变量 V 必须是向量
    'error,V,must be row vector',break
end
for i = 1:n-1                          %循环语句,对向量 V 的元素逐个求最大公约数
    gv = gcd(V(1),V(i+1));
    V(1) = gv;
end
gv;                                    %用默认变量显示结果
```

【例 7-19】 求向量V = [121 143 242 2585 1881 1221 25938]的最大公约数

解:

在命令窗口输入如下程序:

```
 >>V = [121 143 242 2585 1881 1221 25938]   %输入向量 V
V  =                                         %程序响应
  Columns 1 through 6
      121     143     242     2585     1881     1221
  Column 7
    25938
>>gcdv(V)                                    %调用最大公约数函数
ans =                                        %最大公约数
    11
```

7.7 最小公倍数 lcm

若某整数，能被两个已知整数除尽，则某整数是已知两整数的公倍数。在诸多公倍数中必有一个最小的，则称之为最小公倍数。最小公倍数在 MATLAB 中的书写格式为

lcm(A,B)

式中，A、B 为正整数数组，且维数相同。lcm(A,B)则返回一个与 A、B 同维的数组，它的元素对应于数组 A、B 元素的最小公倍数。现举例说明如下。

【例 7-20】 已知向量 ***A*** = [3 15 7 121]，向量 ***B*** = [5 40 49 77]，求 A、B 的最小公倍数向量。

解：

```
>>A = [3 15 7 121];
>>B = [5 40 49 77];
>>lcm(A,B)
ans =
    15      120      49      847
```

【例 7-21】 已知向量 ***V*** = [1 2 3 4]，由此构成范德蒙矩阵 ***A***，另设 ***B*** 为 4 阶魔方矩阵，求 ***A***、***B*** 的最小公倍数矩阵。

解：

```
>>V = [1 2 3 4];              %已知向量 V
>>A = vander(V)               %由向量 V 构成范德蒙矩阵 A,它的每一元素 A(i,j) = V(i)^
                                (n - j)
A =
     1     1     1     1
     8     4     2     1
    27     9     3     1
    64    16     4     1
>>B = magic(4)                %4 阶魔方矩阵
B =
    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1
>>lcm(A,B)                    %矩阵 A、B 的最小公倍数矩阵
ans =
    16     2     3    13
    40    44    10     8
    27    63     6    12
    64   112    60     1
```

以上讨论的是两个数组的最小公倍数，对于多个数值或多个数组的最小公倍数，则 MATLAB 没有内装函数可利用，但是可通过编制程序来解决。现以求多个数值最小公倍数为例，在 M 文件编辑器中输入下列程序，程序名即为函数名 lcmv.m。将此程序存入当前工作目录，即可在命令窗口中调用此函数运行，求解向量中元素的最小公倍数。

```
%本程序计算向量内元素的最小公倍数          %程序注解
function lv = lcmv(V)                     %函数定义项,lcmv 为函数名,V 为输入向量
[m,n] = size(V);                          %测试向量 V 的大小
if m ~ = 1                                %若行数不是 1,则显示出错,指出变量 V 必须是向
                                           量
    'error,V must be row vector', break
end
for i = 1:n - 1                           %循环语句,对向量 V 的元素,逐个求最小公倍数
    lv = lcm(V(1),V(i + 1));
    V(1) = lv;
end
lv;                                       %用默认变量显示结果
```

【例 7-22】 已知向量 $\boldsymbol{V}$ = [30 105 385 102]，求向量各元素的最小公倍数。

解：

```
>>V = [30 385 195 102]                    %输入已知向量 V
V =
    30   385   195   102
>>lcmv (V)                                %计算向量 V 的最小公倍数
    ans =
    510510
```

7.8 数学表达式的化简

MATLAB 中数学表达式的化简是一个有用的工具，它改变了过去用试算法化简数学表达式，从而实现了化简数学表达式的机械化。MATLAB 中的化简函数为 simple 和 simplify，这两者的功能相似，但在实际使用中，前者更有效。数学表达式的化简是通过使用展开函数(expand)、合并函数（collect 或 combine)、因式分解函数（factor）来实现数学表达式的化简。应该指出，机械化的化简并不始终有效，有时还得用人脑的帮助。如果在化简函数后加上分号（;）或在化简函数等式前赋以变量名，则分解过程不显示，否则将显示出较长的简化过程。

数学表达式的化简函数是属于符号数学工具箱（Symbolic Math Toolbox），所以在化简前必须要把数学表达式的变量转换成符号变量。符号变量的建立是十分必要的，因为数学公式的演算、推导和证明需要用符号变量，而不能用个别数字来证明数学公式的成立。创建符号标量、符号变量或符号对象的函数为

```
S = sym(A)
x = sym('x')
```

x = sym('x','real')

x = sym('x','unreal')

S = sym (A) 是创建对象的符号变量，若 A 是字符串，则结果是符号标量或符号变量。若 A 是标量或矩阵，则结果是数值的符号表达式。

对于 x = sym('x'),则创建名称为'x'的符号变量。

对于 x = sym('x','real'),则假定符号变量是实数型。

对于 x = sym('x','unreal'),则对符号变量没有附加特性的纯粹符号变量。

由于符号函数 sym 仅能对一个变量进行转换，对于将多个变量更快捷地创建符号变量，是用符号函数 syms，它能对多个变量同时转换成符号变量。它的书写格式为

syms arg1 arg2 arg3…

syms arg1 arg2 arg3…real

syms arg1 arg2 arg3…unreal

式中，arg1、arg2、arg3…为变量 1、变量 2、变量 3……的符号。后 2 项表明符号变量为实数型还是非实数型。下面举例予以说明。

【例 7-23】 求 4 阶魔方矩阵的符号表达式及其行列式的符号值。

解:

```
>>A = magic(4)              %4 阶魔方矩阵
A =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

>>S = sym(A)                %4 阶魔方矩阵的符号表达式
S =

 [ 16   2   3  13 ]
 [  5  11  10   8 ]
 [  9   7   6  12 ]
 [  4  14  15   1 ]

>>det(A)                    %矩阵 A 的行列式值
ans =
     0
>>s1 = det(S)               %符号矩阵 S 的行列式符号值,在工作空间中显示 s1 是符号变量
s1 =
     0
>>s2 = eval(s1)             %将符号值转换成标量,在工作空间中显示 s2 为数值型
s2 =
```

0

【例 7-24】 已知符号变量 a_{11}、a_{12}、a_{13}、a_{21}、a_{22}、a_{23}、a_{31}、a_{32}、a_{33}构成符号矩阵 $\boldsymbol{A}=\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$，求行列式的符号值。若 $a_{21}=a_{32}=a_{31}=1$，$a_{13}=0$，求其符号逆矩阵 $\boldsymbol{B}$。

解:

```
≫syms a11 a12 a13 a21 a22 a23 a31 a32 a33     %设置符号变量
≫A = [a11 a12 a13;a21 a22 a23;a31 a32 a33]  %符号矩阵 A
A =
[a11,a12,a13]
[a21,a22,a23]
[a31,a32,a33]
≫det(A)                                      %符号矩阵 A 的行列式表达式
ans =
a11* a22* a33 - a11* a23* a32 - a21* a12* a33 + a21* a13* a32 + a31* a12* a23 - a31* a13* a22
≫A1 = [a11 a12 0;1 a22 a23;1 1 a33]
A1 =                                         %对符号矩阵 A,令 a13 = 0,a21 = a32 = a31 =
                                              1,得符号矩阵 A1
[a11, a12,   0]
[  1, a22, a23]
[  1,   1, a33]
≫B = inv(A1)                                 %符号矩阵 A1 的符号逆矩阵
B =
[-(-a22* a33 + a23)/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  -a12* a33/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  a12* a23/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23)]
                                             %符号矩阵 B 的第 1 行
[(-a33 + a23)/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  a11* a33/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  -a11* a23/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23)]
                                             %符号矩阵 B 的第 2 行
[-(-1 + a22)/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  (-a11 + a12)/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23),
  -(-a11* a22 + a12)/(a11* a22* a33 - a11* a23 - a12* a33 + a12* a23)]
                                             %符号矩阵 B 的箱 3 行
```

MATLAB 中的化简函数，主要有两个，即 simplify 和 simple。simplify 是使用 MAPLE（数学软件）的简化规则，而 simple 则使化简后的表达式字符长度最短。simplify 的书写格式如下：

R = simplify(S)

式中，S 为待简化的符号矩阵，simplify (S)对其中每一个元素进行化简。

下面举例说明它的应用。

【例 7-25】 试化简分式多项式 $\frac{2x-4}{x^2-5x+6}+\frac{x-1}{x^2-4x+3}$ (7-1)

及多项式 $(x-1)^4+4(x-1)^3+6(x-1)^2+(x-2)$ (7-2)

解：

```
>>syms x                                                          %设置符号变量
>>y1 = simplify((2*x-4)/(x^2-5*x+6)+(x-1)/(x^2-4*x+3))            %简化式(7-1)
>>y1
y1 =                                                              %简化结果
3/(x-3)
>>y2 = simple((x-1)^4+4*(x-1)^3+6*(x-1)^2+(x-2))                  %简化式(7-2)
>>y2
y2 =                                                              %简化结果,由此说明,
                                                                   通过化简,多项式的
                                                                   书写长度大大缩短了

   x^4-3*x+1
```

【例 7-26】 试用化简函数 simple 化简下列三角函数表达式 $y=\sin^2 x+3\sin x\cos x+5\cos^2 x$。

解：

```
>>syms x                                                          %设置符号变量
>>y = sin(x)^2+3*sin(x)*cos(x)+5*cos(x)^2;                        %输入三角函数表达式
>>y = simple(y)                                                   %调用化简函数
y =                                                               %化简结果
3+2*cos(2*x)+3/2*sin(2*x)
```

7.9 数组的平均值 mean 及标准偏差 std

平均值与标准偏差在日常生活中受到广泛的应用。它是最基本的统计指标之一。如产品的平均价格、汽车的平均车速、企业职工的平均工资、学生考试的平均成绩和人体的平均身高与体重等。标准偏差则用来衡量一组数据偏离平均值的程度。标准偏差越大，则说明向量元素的离散程度越大。在 MATLAB 中，数组的平均值及标准偏差的书写格式如下：

```
mean(X)
mean(X,dim)
std(A)
std(A,dim)
```

若 X 为向量，则 mean (X)返回一个标量，它等于所有元素之和除以向量的列数。

若 X 为矩阵，则 mean (X)返回一个行向量，它的元素对应所在列的平均值。

标量 dim 指矩阵的维数，dim = 1，对应所在列的平均值，dim = 2，则对应所在行的平均值。

若 A 为向量，则 std (A)返回一个标量（也称为标准差），它等于

$$(\sum(a_i - a_v)^2)/(n-1)$$

式中，a_i 为向量 A 的元素，i 从 1：n，n 为向量 A 的列数；a_v 为向量 A 元素的平均值。

若 A 为矩阵，则 std (A)返回一个行向量，它的每一个元素对应于 A 矩阵列向量的标准差。

关于 dim 的含义则与前述相同。现举例予以说明。

【例 7-27】 已知学生考试成绩表如向量 ***X*** = [96 84 66 90 72 78]，求总分、平均分和标准差。

解：

```
>>X = [96 84 66 90 72 78]        %已知考试成绩向量 X
X =
    96    84    66    90    72    78
>>sum(X)                         %总分
ans =
  486
>>mean(X)                        %平均分
ans =
  81
>>std(X)                         %考试成绩的标准差
ans =
  11.2250
```

【例 7-28】 今有 8 个学生的考试成绩如矩阵 X，学生以列排列，各门课程成绩以行排列，求 8 位学生的总分、平均分和标准差。以列数表示学生号，问哪位学生总分最高，哪位平均分最高，哪位分数离散性最小。

X =

```
    73    92    76    68    44    71    74   100
    55    74   100    57    67    78    90    66
    81    79    82    74    73    71    92    84
    84    77    86    49    31    71    90    82
    62    72   100    80    33    77    99    60
    97    85    85    94    63    72    90    88
```

解：

```
>>X = [73   92   76   68   44   71   74100;          %输入学生考试成绩矩阵
       55   74  100   57   67   78   90  66;
       81   79   82   74   73   71   92  84;
       84   77   86   49   31   71   90  82;
```

```
      62   72  100   80   33   77   99  60;
      97   85   85   94   63   72   90  88; ]
X =                                % MATLAB 输出响应
    73    92    76    68    44    71    74   100
    55    74   100    57    67    78    90    66
    81    79    82    74    73    71    92    84
    84    77    86    49    31    71    90    82
    62    72   100    80    33    77    99    60
    97    85    85    94    63    72    90    88
  ≫sum(X)                        %学生考试成绩总分
  ans =
    452   479   529   422   311   440   535   480
  ≫mean(X)                       %平均分
  ans =
    75.3333   79.8333   88.1667   70.3333   51.8333   73.3333   89.1667   80.0000
  ≫std(X)                        %标准差
  ans =
    15.3188   7.4677   9.8065   16.1576   18.1815   3.2660   8.2077   14.6969
```

由此可见，7 号学生总分最高为 535 分，平均分也最高为 89.1667（这两者是一致的）考试成绩离散度最小为 6 号学生，标准差为 3.2660，他的考分集中在 73 分左右。

7.10 数组元素的最大 max 和最小 min

寻找数组元素的最大值和最小值是常用的函数。例如用消元法解线性方程组时，寻找主元时，就要找矩阵元素中的最大值。寻找最短路径时要用到最小值函数。寻找测试成绩总分最高者亦需要用最大值函数。MATLAB 中对于寻找数组中元素最大值和最小值的函数的书写格式如下：

```
C = max(A)
C = max(A,B)
C = max(A,[],dim)
[C,I] = max(A)
```

假如 A 是向量，则 max (A)返回一个向量 A 中的最大值元素。

假如 A 是矩阵，则 max (A)返回一个行向量，行向量的元素是对应列元素中的最大值。

dim 为维数，用来确定寻找数组中的哪一维的最大值。dim = 1 与 max (A)相同。dim = 2 则返回一个列向量，列向量的元素是对应行元素中的最大值。必需指出 C = max(A,[],dim) 中的空格符[]不能省，否则将出错。

[C,I] = max(A)，则显示最大值元素和最大值元素所处的下标位置。C 显示矩阵列元素的最大值，I 则显示最大值元素所处的下标。

C = max(A,B)，则显示一个与 A、B 同维的矩阵，取 A，B 矩阵中元素为最大。

下面举例予以说明。

【例 7-29】　已知向量$\boldsymbol{A}$ = [16 2 3 13 5 11 10 8]，求最大值元素。

解：

```
>>A = [16 2 3 5 11 10 8]
A =
     16     2     3     5     11     10     8
>>max(A)
ans =
     16
```

【例 7-30】　已知下列矩阵，求最大值元素及所处位置。

```
  95    23    60    48    89    76    45     1
 164    88   123   158   184   147    35    81
 280   275   123   268    17   105   243     2
  55    81    79   241   108    79     6   298
 222   465   232   209   423   262   101   336
 502    11   408   227   499   301   425   257
 213   132   135   477   211   379   105   488
 302   688   682   474   397   719   657   515
```

解：

```
>>A = [95    23    60    48    89    76    45     1;      %输入已知矩阵 A
      164    88   123   158   184   147    35    81;
      280   275   123   268    17   105   243     2;
       55    81    79   241   108    79     6   298;
      222   465   232   209   423   262   101   336;
      502    11   408   227   499   301   425   257;
      213   132   135   477   211   379   105   488;
      302   688   682   474   397   719   657   515]
A  =
       95    23    60    48    89    76    45     1
      164    88   123   158   184   147    35    81
      280   275   123   268    17   105   243     2
       55    81    79   241   108    79     6   298
      222   465   232   209   423   262   101   336
      502    11   408   227   499   301   425   257
      213   132   135   477   211   379   105   488
      302   688   682   474   397   719   657   515
>>[C,I] = max(A)                                    %寻找列最大值元素
C =
```

```
       502   688   682   477   499   719   657   515
I =
         6     8     8     7     6     8     8     8
>>[C1,I1] = max(C)                  %在列最大值中再寻找最大值
C1 =                                %矩阵 A 的最大值元素为 719,所处位置为第 8 行第 6 列
   719
I1 =
     6
```

【例 7-31】 已知矩阵 ***A*** 为 4 阶单位方阵，矩阵 $\boldsymbol{B}=\begin{bmatrix}0 & 0 & 0 & 0\\0 & 100 & 100 & 0\\0 & 100 & 100 & 0\\0 & 0 & 0 & 0\end{bmatrix}$，求C = max(A,B)。

解:

```
>>A = ones(4)                                      %已知矩阵 A
A =
     1     1     1     1
     1     1     1     1
     1     1     1     1
     1     1     1     1
>>B = [0 0 0 0;0 100 100 0;0 100 100 0;0 0 0 0]    %已知矩阵 B
B =
     0     0     0     0
     0   100   100     0
     0   100   100     0
     0     0     0     0
>>C = max(A,B)                                     %取矩阵 A、B 中最大者
C =
     1     1     1     1
     1   100   100     1
     1   100   100     1
     1     1     1     1
```

对于数组的最小值函数 min，则与数组的最大值函数 max 十分相似，故不再重复说明和举例。其书写格式如下：

```
C = min(A)
C = min(A,B)
C = min(A,[],dim)
[C,I] = min(A)
```

7.11 多边形面积的计算

多边形面积的计算，在初等数学中是常见的问题。如三角形面积、多边形面积等。MATLAB 提供一般性多边形面积计算函数，它的书写格式如下：

A = polyarea(X,Y)

A = polyarea(X,Y,dim)

若 X、Y 是同维向量，并且 X、Y 的对应项表示多边形的每一个顶点坐标，其次多边形的起点和终点必须重合，则A = polyarea(X,Y)为多边形的面积。

若 X、Y 为同维矩阵，则A = polyarea(X,Y)为 X、Y 对应列所构成多边形面积。对应列亦需满足是多边形的每一个顶点坐标，且起点和终点必须重合。

在计算面积中，多边形的边（顶点除外）不能相交，若相交，则算出的面积为顺时针所环绕面积与逆时针所环绕面积之差。下面举例予以说明。

【例 7-32】 计算不对称五边形（见图 7-1）的面积，其各顶点的坐标为 a(0,0)、b(−1,3)、c(4,6)、d(7,3)和 e(5,0)。

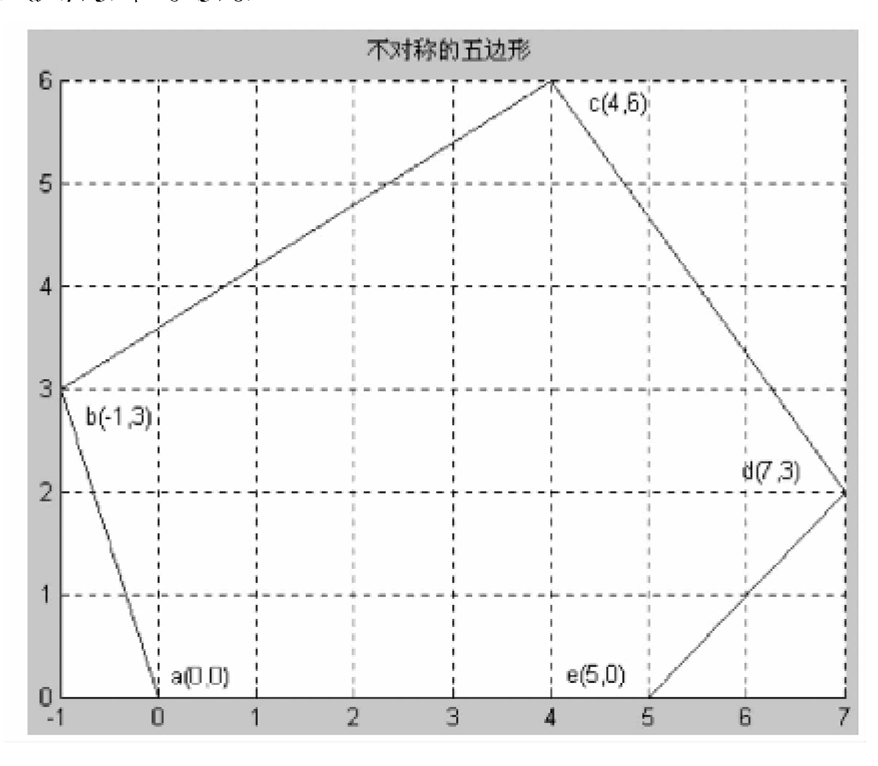


图 7-1 不对称的五边形的面积计算

解：

在 MATLAB 命令窗口中输入如下程序：

```
≫X = [0 -1 4 7 5];            %将各顶点 a、b、c、d、e 的横坐标写入 X
≫Y = [0 3 6 3 0];             %将各顶点 a、b、c、d、e 的纵坐标写入 Y
≫S5 = polyarea(X,Y)           %计算五边形的面积
S5 =
  31.5000
```

【例 7-33】 计算不规则七边形（见图 7-2）的面积。各顶点的坐标为 a_1(−1.2,0)、a_2

(-0.5,5.5)、a_3(-1.5,5)、a_4(-1.6,6.5)、a_5(0,8)、a_6(1,7)、a_7(1.2,0)。

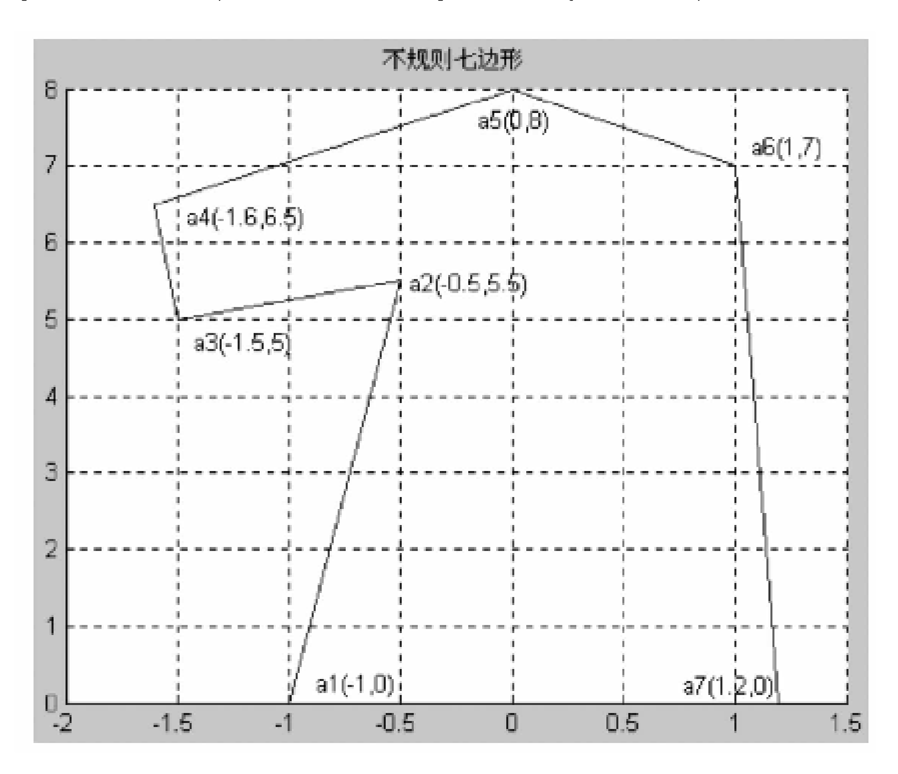


图 7-2　计算不规则 7 边形面积

解:

在 MATLAB 命令窗口中输入如下程序:

```
≫X1 = [-1 -0.5 -1.5 -1.6 0 1 1.2];      %将各顶点 a1、a2、a3、a4、a5、a6、
                                          a7 的横坐标写入 X1
≫Y1 = [0 5.5 5 6.5 8 7 0];              %将各顶点 a1、a2、a3、a4、a5、a6、a7 的纵坐标
                                          写入 Y1
≫S7 = polyarea(X1,Y1)                    %计算七边形面积
S7 =
    15.3500
```

【例 7-34】　计算单位圆内，内接正 12 边形、120 边形和 1200 边形的面积。

解:

在 MATLAB 命令窗口中输入如下程序:

```
≫n = 12;                          %设正多边形数为 12
≫x = 0:2* pi/n:2* pi* (n-1)/n;    %将单位圆 n 等分,等分点则成为正多边形的顶点
≫X = cos(x);                      %正多边形顶点的横坐标向量
≫Y = sin(x);                      %正多边形顶点的纵坐标向量
≫S12 = polyarea(X,Y)              %单位圆内,内接正 12 边形的面积
S12 =
    3
```

```
>>n = 120;                              %设正多边形数为 120
>> x = 0:2* pi/n:2* pi* (n-1)/n;        %将单位圆 n 等分,等分点则成为正多边形的顶点
>>X = cos(x);                           %正多边形顶点的横坐标向量
>>Y = sin(x);                           %正多边形顶点的纵坐标向量
>>S120 = polyarea(X,Y)                  %单位圆内,内接正 120 边形的的面积
S120 =
    3.1402
>>format long
>>n = 1200;                             %设正多边形数为 1200
>>x = 0:2* pi/n:2* pi* (n-1)/n;         %将单位圆 n 等分,等分点则成为正多边形的顶点
>>X = cos(x);                           %正多边形顶点的横坐标向量
>>Y = sin(x);                           %正多边形顶点的纵坐标向量
>>S1200 = polyarea(X,Y)                 %单位圆内,内接正 1200 边形的的面积。这时多边形面
                                          积,已与圆面积十分接近
S1200 =
    3.14157829885175
>>pi                                    %精确的 pi 值
ans =
    3.14159265358979
```

【例 7-35】 已知矩阵 $\boldsymbol{X}=\begin{bmatrix}4 & 8 & 12\\3 & 6 & 9\\0 & 0 & 0\end{bmatrix}$; $\boldsymbol{Y}=\begin{bmatrix}0 & 0 & 0\\2 & 4 & 6\\2 & 4 & 6\end{bmatrix}$, 求 3 个三角形面积。

解:

```
>>X = [4 8 12;3 6 9;0 0 0]              %输入矩阵 X
X =
    4    8   12
    3    6    9
    0    0    0
>>Y = [0 0 0;2 4 6;2 4 6]               %输入矩阵 Y
Y =
    0    0    0
    2    4    6
    2    4    6
>>polyarea(X,Y)                         %计算由 X、Y 的列向量所构成的 3 个三角形面积
ans =
    3      12      27
>>plot(X,Y)                             %绘制矩阵 X、Y 所构成线性图,如图 7-3 所示
```

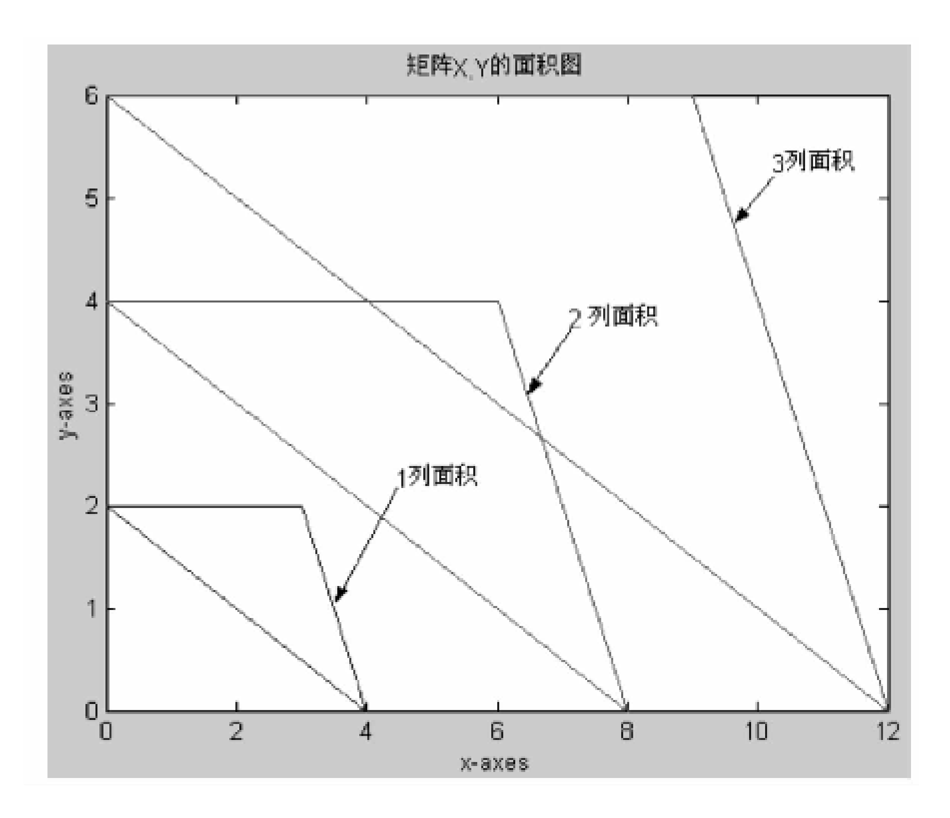


图 7-3 矩阵 X、Y 的线型图

7.12 符号表达式的求和函数 symsum

MATLAB 中符号表达式的求和函数 symsum 是有用的工具。它可以用来计算各种常用级数的有限项之和或无限项之和。符号表达式的求和函数的书写格式如下：

r = symsum(s)

r = symsum(s,v)

r = symsum(s,a,b)

r = symsum(s,v,a,b)

式中，symsum (s)是用来计算符号表达式 s 的总和。s 是与符号变量 k 有关表达式，k 从 0 到 k－1 项计算 s 的总和。函数 symsum 一词，是由 sym （符号的缩写） 和 sum （求和的缩写） 合并而成。

symsum (s,v)是计算与符号变量 v 有关的符号表达式 s 之和。符号变量 v 从 0 到 v－1 项。

symsum (s,a,b)是用来计算符号表达式 s 的总和。s 是与符号变量 k 有关表达式，但 k 从 a 到 b 计量。

symsum (s,v,a,b)是用来计算符号表达式 s 的总和。s 是与符号变量 v 有关表达式，但 v 从 a 到 b 计量。下面举例予以说明。

【例 7-36】 已知等差级数的首项 a、公差 d 和项数 n，求 n 项级数之和 s。

解：

```
>>syms a d n                          %设置符号变量
>>(v = a + (n - 1) * d);              %级数一般项表达式,n = 1 为首项
>>s = simple(symsum(v,1,n))           %求 1 到 n 项级数之和,并加以化简
s =                                   %等差级数 n 项之和的计算公式
```

```
1/2* n* (2* a - d + d* n)
```

【例 7-37】 求下列级数之和：

$$s_1 = 1 + 2 + 3 + \cdots + (n-1) + n \tag{7-3}$$

$$s_2 = 1^2 + 2^2 + 3^2 + \cdots + (n-1)^2 + n^2 \tag{7-4}$$

$$s_3 = 1^3 + 2^3 + 3^3 + \cdots + (n-1)^3 + n^3 \tag{7-5}$$

解：

```
>>syms k n                              %设置符号变量
>>s1 = simple(symsum(k,1,n))            %经过化简、求和，计算式(7-3)
  s1 =
      1/2* n* (n + 1)
>>s2 = simple(symsum(k^2,1,n))          %经过化简、求和，计算式(7-4)
  s2 =
      1/6* n* (n + 1)* (2* n + 1)
>>s3 = simple(symsum(k^3,1,n))          %经过化简、求和，计算式(7-5)
  s3 =
      1/4* n^2* (n + 1)^2
```

【例 7-38】 求无限项级数

$$s_4 = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \cdots + (-1)^{(n+1)}\frac{1}{2n-1} + \cdots \tag{7-6}$$

之和。

解：

```
>>syms s n                              %设置符号变量
>>s = (-1)^(n + 1)* (1/(2* n - 1));     %级数的一般项表达式
>>s4 = simple(symsum(s,1,inf))          %经过化简、求和，计算式(7-6)，项数由第 1 项至
                                          无限项(inf)
s4 =                                    %级数和为 π/4
1/4* pi
```

【例 7-39】 计算等比级数的 n 项之和，首项为 a，公比为 r，项数为 n。若 $a=2$、$r=1/3$，求 n 项级数之和。

解：

```
>>syms a r n                            %设符号变量
>>s = a* r^(n-1);                       %级数一般项表达式
>>s5 = simple(symsum(s,1,n))            %求 n 项级数之和
s5 =                                    % MATLAB 得不出一般解，实际上 s5 等于
```

$a\dfrac{1-r^{n+1}}{1-r}$，所以对 MATLAB 的计算结果，有时需要进行核对

```
a* sum(r^n/r,r = 1..n)
>>a = 2; r = 1/3;                       %但是当 a、r 为数值时
```

```
≫s = a* r.^(n-1);                      %级数一般项表达式
≫s6 = simple(symsum(s,1,n))            %求 n 项级数之和
s6 =                                   %当 n > 1 时,级数求和表达式成立
-3^(1-n)+3
```

7.13 数组的取整函数

在分析计算工作中，经常需要对计算结果取整数。如产品数、动物头数、某事件出现次数等。

因为这类数字不可能以分数或小数形式出现。但是取整亦需有规则，例如数字 5.5，取整后是取 5 还是取 6 呢。

MATLAB 设置了 4 种数组取整函数，它对数组中每一个元素都进行取整，对于复数，则分别对复数的实部和虚部予以取整。这 4 种取整函数是

fix(X)——沿着零取整

round(X)——沿着最近的整数取整

ceil(X)——沿着正 inf 取整

floor(X)——沿着负 inf 取整

下面举例予以说明。

【例 7-40】 已知向量 $\boldsymbol{a}=[1.4\ 1.5\ -1.4\ -1.5]$，求沿零取整（fix）、就近取整（round）、沿正 inf 取整（ceil）和沿负 inf 取整（floor）。

解：

```
≫a = [1.4 1.5 -1.4 -1.5]
a =
    1.4000    1.5000    -1.4000    -1.5000
≫fix(a)
ans =
    1    1    -1    -1
≫round(a)
ans =
    1    2    -1    -2
≫ceil(a)
ans =
    2    2    -1    -1
≫floor(a)
ans =
    1    1    -2    -2
```

【例 7-41】 已知 3 行 4 列均匀分布的随机矩阵 $\boldsymbol{A}$，求 $100\boldsymbol{A}$ 后沿零取整、就近取整、沿 +inf 取整和沿 -inf 取整。$\boldsymbol{A}$ 矩阵为

$$\begin{bmatrix} 0.9501 & 0.4860 & 0.4565 & 0.4447 \\ 0.2311 & 0.8913 & 0.0185 & 0.6154 \\ 0.6068 & 0.7621 & 0.8214 & 0.7919 \end{bmatrix}$$

解:

```
>>A = rand(3,4)
A  =
    0.9501   0.4860   0.4565   0.4447
    0.2311   0.8913   0.0185   0.6154
    0.6068   0.7621   0.8214   0.7919
>>B = 100* A
B =
    95.0129    48.5982    45.6468    44.4703
    23.1139    89.1299     1.8504    61.5432
    60.6843    76.2097    82.1407    79.1937
>>fix(B)                                    %沿零取整
ans =
      95     48     45     44
      23     89      1     61
      60     76     82     79
>>round(B)                                  %就近取整
ans =
      95     49     46     44
      23     89      2     62
      61     76     82     79
>>floor(B)                                  %沿 inf 取整
ans =
      95     48     45     44
      23     89      1     61
      60     76     82     79
>>ceil(B)                                   %沿 - inf 取整
ans =
      95     49     46     45
      24     90      2     62
      61     77     83     80
```

由此说明，矩阵 B 采取 4 种取整函数后，由于取整的方向不同，所以其结果稍有不同。

7.14　数组的模数 mod

两个实数数组 ***X***、***Y***，数组相除后的余数数组称为模数，它的书写格式为

mod(X,Y)

它相当于X - Y* floor(X./Y)，假如 Y 的某个元素 y 为零，则约定mod(X,0) = x ，式中 x 为 X 数组中的元素。

【例 7-42】 已知***X*** = [97 23 33 92 96]、***Y*** = [16 7 5 11 13]，求mod(X,Y)。

解：

```
≫X = [97 23 33 92 96];
≫Y = [16 7 5 11 13];
≫mod(X,Y)
ans =
         1      2      3      4      5
```

【例 7-43】 已知矩阵 ***X*** 为 4 阶魔方阵，矩阵 $\boldsymbol{Y}=\begin{bmatrix}2 & 3 & 4 & 7\\ 11 & 13 & 17 & 19\\ 2 & 4 & 6 & 8\\ 23 & 29 & 31 & 34\end{bmatrix}$，求mod(X,Y)。

解：

```
≫X = magic(4)
X =

    16     2     3    13
     5    11    10     8
     9     7     6    12
     4    14    15     1

≫Y = [2 3 4 7; 11 13 17 19;2 4 6 8;23 29 31 34]
Y =

     2     3     5     7
    11    13    17    19
     2     4     6     8
    23    29    31    34

≫mod(X,Y)
ans =

     0     2     3     6
     5    11    10     8
     1     3     0     4
     4    14    15     1
```

7.15 不定方程组的整数解

不定方程组的整数解，在初等数学中是常见的。所谓不定方程组，是指方程式的数量小于方程式的变量数。因此有无穷多个解。但若附加条件，满足最小整数条件，则方程组就有解了。

【例 7-44】 某整数被 3 除余 1，被 5 除余 3，被 7 除余 5，被 11 除余 7，求该数的最小正整数。根据题意可列出方程组如下：

$$n/3 = a + 1/3$$
$$n/5 = b + 3/5$$
$$n/7 = c + 5/7$$
$$n/11 = d + 7/11$$

式中，n 为待求整数，a、b、c、d 被 3、5、7、11 除后的商数。

由于该方程组中含有 5 个未知数，而方程式只有 4 个，故为不定方程组，有无穷个解。但可以用循环语句搜索符合余数条件的最小整数解。程序名为 intsolve.m，在 M 文件编辑窗口中输入如下程序：

```
%不定方程组的整数解
for n=1:1000                          %设 n 为待求解,从 1:1000 进行搜索
    if mod(n,3)==1&mod(n,5)==3&mod(n,7)==5&mod(n,11)==7
                                      %满足所有余数的条件,则中断循环
        break
    end
end,n                                 %显示整数解
a=(n-mod(n,3))/3                      %显示商数
b=(n-mod(n,5))/5
c=(n-mod(n,7))/7
d=(n-mod(n,11))/11
```

在 MATLAB 命令窗口运行该程序,结果如下：

```
>>intsolve                            %调用解本题的程序
n=                                    %最小整数解的答案
    733
a=                                    %被 3 除后的商
    244
b=                                    %被 5 除后的商
    146
c=                                    %被 7 除后的商
    104
d=                                    %被 11 除后的商
    66
```

为了核对解题结果的正确性,在命令窗口输入如下程序：

```
>>mod(733,3)
ans=
    1
>>mod(733,5)
```

```
ans =
      3
≫mod(733,7)
ans =
      5
≫mod(733,11)
ans =
      7
```

复核结果，解题是正确的。由于搜索 n 从 1 开始，n = 733 是满足所有余数条件的最小数。

7.16 变量替换函数 subs

符号变量用常数替换或用新的符号变量替换是经常用到的。MATLAB 中，符号变量替换函数的书写格式为

subs(s)

subs(s,old,new)

式中，s 为符号变量表达式；“old”为表达式 s 中的旧变量；“new”为表达式中的新变量。subs (s)用于替换已经赋值的常数变量。subs(s,old,new)则用新的变量替换旧的变量。注意在旧变量上必须加单引号，现举例如下：

【例 7-45】 已知 $a = 3$、$b = 4$，求 $s = a\sin x/\sqrt{a^2 + b^2} + b\cos x/\sqrt{a^2 + b^2}$ 的替换表达式。

解：

```
≫syms a b x                              %设置符号变量
≫s = a/((a^2 + b^2)^(1/2))* sin(x) + b/((a^2 + b^2)^(1/2))* cos(x)
                                          %写入符号表达式 s
s =
a* sin(x)/(a^2 + b^2)^(1/2) + b* cos(x)/(a^2 + b^2)^(1/2)
≫a = 3,b = 4                              %代入常数变量的值
a =
      3
b =
      4
≫s = subs(s)                              %用常数替代常数变量 a、b
s =
3/25* sin(x)* 25^(1/2) + 4/25* cos(x)* 25^(1/2)
≫s = eval(s)                              %将字符串数字转换成数值。由此得常数替换后
                                            的表达式 s
s =
3/5* sin(x) + 4/5* cos(x)
```

【例 7-46】 已知多项式符号表达式 $s = x^5 + 10x^4 + 40x^3 + 80x^2 + 80x + 32$，试用新变量 $u = x + 2$ 替换旧变量 x。

解：

```
≫syms x u                                      %设置符号变量
≫s = x^5 + 10* x^4 + 40* x^3 + 80* x^2 + 80* x + 32   %输入多项式符号表达式
s =
x^5 + 10* x^4 + 40* x^3 + 80* x^2 + 80* x + 32
≫s = subs(s, x, u - 2)                          %变量替换,用 u - 2 替换 x
s =
(u-2)^5 + 10 * (u - 2)^4 + 40* (u - 2)^3 + 80* (u - 2)^2 + 80* u - 128
≫s = simple(s)                                  %化简变量替换后的多项式表达式
s =
u^5
```

7.17 平面几何的证明题

在平面几何学中，有关证明的命题是相当多的。众所周知，如三角形三内角之和为180°。三角形两边之和大于第三边。三角形的3个顶点到对边的3根垂直线交于一点等。这些证明题都是通过定义和演算取得的。有时还得增添辅助线。但是能否通过 MATLAB 来帮助完成证明题的任务。回答是肯定的。下面举例予以说明。

【例 7-47】 证明下列四边形面积

$$S = \frac{1}{2} d_1 d_2 {}^{*} \sin\zeta$$

式中 d_1、d_2——四边形的对角线长度；

ζ——两对角线 d_1、d_2 的夹角。

计算公式是否成立

证明：

将四边形放入直角坐标中，四边形的一个顶点置于原点，一边与 x 轴重合，如图 7-4 所示。在命令窗口中输入如下程序：

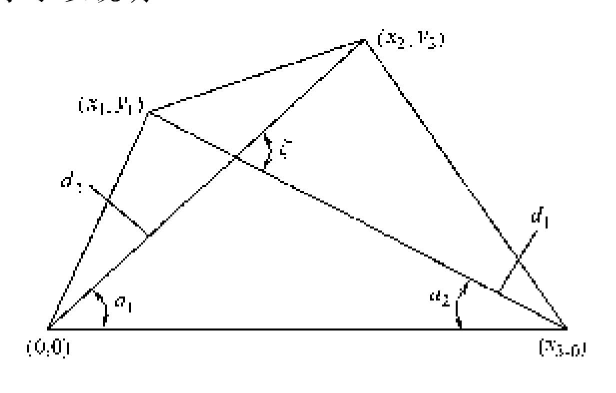


图 7-4 四边形的面积计算

```
≫syms x1 x2 x3 y1 y2 d1 d2 a1 a2               %设置符号变量
≫X = [0 x1 x2 x3 0];                            %设置四边形的横坐标向量
≫Y = [0 y1 y2 0 0];                             %设置四边形的纵坐标向量
≫s = polyarea(X, Y)                             %计算四边形面积
s =
abs( - 1/2* x1* y1 - 1/2* (x2 - x1)* (y2 + y1) - 1/2* (x3 - x2)* y2)
≫s = simple(s)                                  %用化简函数,化简面积表达式 s
s =
abs( - 1/2* x2* y1 + 1/2* x1* y2 - 1/2* y2* x3)
≫s = 1/2* abs(x2* y1 + x3* y2 - x1* y2);        %提取公因子 1/2
```

```
≫s = subs(s,x1,x3 - d1* cos(a2))                %用 x3 - d1* cos(a2)替代 x1
s =
1/2* abs(x2* y1 - (x3 - d1* cos(a2))* y2 + y2* x3)
≫s = subs(s,x2,d2* cos(a1))                     %用 d2* cos(a1)替代 x2
s =
1/2* abs(d2* cos(a1)* y1 - (x3 - d1* cos(a2))* y2 + y2* x3)
≫s = subs(s,y1,d1* sin(a2))                     %用 d1* sin(a2)替代 y1
s =
1/2* abs(d2* cos(a1)* d1* sin(a2) - (x3 - d1* cos(a2))* y2 + y2* x3)
≫s = subs(s,y2,d2* sin(a1))                     %用 d2* sin(a1)替代 y2
s =
1/2* abs(d2* cos(a1)* d1* sin(a2) - (x3 - d1* cos(a2))* d2* sin(a1) + d2* sin(a1)* x3)
≫s = simple(s)                                  %再次化简四边形面积表达式
s =
1/2* abs(d1* d2* sin(a1 + a2))
≫s = subs(s,a1 + a2,zeta)                       % 由于 zeta = a1 + a2,所以再替代,即得所
                                                  要证明的四边形面积公式
      s =
        1/2* abs(d1* d2* sin(zeta))
```

由本例说明，使用 MATLAB，用户几乎可以不假思索，就可以证明四边形的面积计算公式。

【例 7-48】 试证明三角形面积

$$A = \sqrt{s(s-a)(s-b)(s-c)}$$

式中，$s = (a+b+c)/2$，a、b、c 为三角形各边的长。

证明：

```
≫syms a b c zeta                                %设置符号变量 a、b、c 和 a、c 两边的夹角 ze-
                                                  ta
≫zeta = acos((a^2 + c^2 - b^2)/(2* a* c))      %由余弦定律得 zeta
zeta =
acos(1/2* (a^2 + c^2 - b^2)/a/c)
≫A = 1/2* a* c* sin(zeta)                      %三角形面积
A =
1/4* a* c* (4 - (a^2 + c^2 - b^2)^2/a^2/c^2)^(1/2)
≫Asq = A^2                                      %三角形面积的平方
Asq =
1/16* a^2* c^2* (4 - (a^2 + c^2 - b^2)^2/a^2/c^2)
≫[n,d] = numden(Asq)                            %提取 Asq 的分子多项式 n 和分母多项式 d

n =
```

```
- a^2 * c^2 * (- 2 * a^2 * b^2 + a^4 + c^4 + b^4 - 2 * a^2 * c^2 - 2 * b^2 * c^2)
d =
16 * a^2 * c^2
>>n = simple(n)                                   %化简分子多项式 n
n =
- a^2 * c^2 * (a + b - c) * (b + a + c) * (a + c - b) * (- c + a - b)
>>Asq = n/d                                       %化简后的 Asq
Asq =
- 1/16 * (a + b - c) * (b + a + c) * (a + c - b) * (- c + a - b)
>>A = sqrt(Asq)                                   %化简后的面积表达式 A
A =
1/4 * (- (a + b - c) * (b + a + c) * (a + c - b) * (- c + a - b))^(1/2)
>>s = (a + b + c)/2                               %设置变量
>>A = (s * (s - a) * (s - b) * (s - c))^(1/2)     %将 s 代入 A 即为所求面积公式
```

【例 7-49】 求证在三角形 ABC 内，3 根高 AD、BE、CF 交于一点 H 如图 7-5 所示。

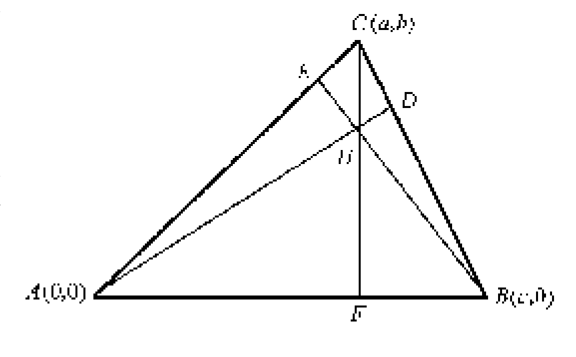


图 7-5　三角形内 3 根高交于一点

证明：

置三角形 ABC 的边 AB 于横坐标上，点 A 在原点上，点 A、B、C 的坐标如图 7-5 所示。AC 的直线方程为

$$y = \frac{b}{a}x \tag{7-7}$$

BC 的直线方程式为

$$y = \frac{b}{(a-c)}(x-c) \tag{7-8}$$

直线 AD 垂直 BC 的方程式为

$$y = \frac{(c-a)}{b}x \tag{7-9}$$

直线 BE 垂直 AC 的方程式为

$$y = -\frac{a}{b}(x-c) \tag{7-10}$$

直线 CF 垂直于 AB 的方程式为

$$x - a = 0 \tag{7-11}$$

在 MATLAB 命令窗口中，列出式（7-9）、式（7-10）、式（7-11）的系数行列式，并检查行列式的值是否为零，若为零，则 3 线共点，亦即三角形 ABC 内 3 个高 AD、BE、CF 公共交于一点。

```
>>syms a b c;                                             %设字符串变量
>>A = [(c - a)/b, - 1, 0; - a/b, - 1, a * c/b; 1, 0, - a]  %三角形 3 根高直线方程式
                                                            的系数矩阵
A =
```

```
[ (c-a) /b,      -1,      0]
   [-a/b,        -1,    a*c/b]
      [1,         0,      -a]
>>det (A)                               %系数行列式的值
ans =                                   %3个直线方程式的系数行
                                          列式为零，则这3根直线
                                          交于一点（参阅数学手
                                          册）
0
```

所以三角形 ABC 内 3 根高交于一点

第 7 章习题

7-1　求 300~500 间有几个素数，并列出素数。

7-2　用 isprime 命令检查正整数 377、601、1171、1571、13577 是否是素数。

7-3　化简代数表达式

$$\frac{2x-4}{x^2-7x+10}+\frac{3x-24}{x^2-13x+40}$$

7-4　化简代数表达式

$$y=(x-3)^3+6(x-3)^2+9(x-3)+1$$

7-5　已知向量 $\boldsymbol{A}$ = [16 17 18 49]、$\boldsymbol{B}$ = [64 34 65 84]，求 $\boldsymbol{A}$、$\boldsymbol{B}$ 间最大公约数。

7-6　求向量 $\boldsymbol{A}$ 元素中最大公约数，$\boldsymbol{A}$ = [1391 5213 7913 11401 2691]。

7-7　求向量 $\boldsymbol{B}$ 元素中最小公倍数，$\boldsymbol{B}$ = [48 96 144 444 644]。

7-8　已知向量 $\boldsymbol{A}_1$ = [16、25、18、19]；$\boldsymbol{B}_1$ = [32、55、99、13]，求 $\boldsymbol{A}_1$、$\boldsymbol{B}_1$ 间最小公倍数。

7-9　exp (1) = 2.71828，求采用取整函数 fix、floor、round 和 ceil 后数值的变化。

7-10　利用符号表达式求和函数 symsum 计算下列级数之和：

(1) $s_1=1^2+3^2+5^2+\cdots+(2n-1)^2$

(2) $s_2=1\times2+2\times3+3\times4+\cdots+(n-1)n$

7-11　某整数被 3 除余 1，被 5 除余 3，被 8 除余 7，被 13 除余 11，求该数的最小正整数。

第 8 章　MATLAB 在微积分中的应用

MATLAB 在矩阵运算方面为用户提供多种函数，使矩阵运算，线性方程组求解带来极大的方便。与此同时，MATLAB 对微分、积分运算，微分方程的符号解、微分方程的数值解、极限的计算、级数的计算、差分方程的求解都提供了多种计算函数，使读者在繁琐的计算中解放出来，使解题成为一种乐趣。

8.1　差分与近似微分

差分函数diff(x)用来计算向量 x 在相邻二元素间差分。它的书写格式为

$$Y = diff(X) \tag{8-1}$$

$$Y = diff(X, n) \tag{8-2}$$

$$Y = diff(X, n, dim) \tag{8-3}$$

式（8-1）中 X、Y 为向量或矩阵，n、dim 为标量。当 X 为向量，则它返回一个 X 的相邻元素的差分向量，它比 X 少一个元素。

$$diff(X) = [X(2) - X(1), X(3) - X(2), \cdots, X(n) - X(n-1)]$$

当 X 为矩阵，则它返回一个矩阵，它的每一列是原来列元素的差分。

式（8-2）则返回向量 X 的 n 次差分。式（8-3）则沿着 dim 所指定的维数进行差分运算。dim 默认时，则沿着第一个非单独维进行差分计算。

【例 8-1】　已知 $\boldsymbol{X} = [1\ 2\ 3\ 4\ 5\ 6]$，求其一次差分向量 $\boldsymbol{y}_1$ 及二次差分向量 $\boldsymbol{y}_2$。

解:

```
>>X = 1:6                          %输入向量 X
X =
     1     2     3     4     5     6
>>y1 = diff(X)                     %一阶差分向量
y1 =
     1     1     1     1     1
>>y2 = diff(X,2)                   %二阶差分向量
y2 =
     0     0     0     0
```

【例 8-2】　已知 $\boldsymbol{X}$ 为 4 阶魔方阵，求其一次沿列的差分向量 $\boldsymbol{y}_1$ 和一次沿行的差分向量 $\boldsymbol{y}_2$。

解:

```
>>X = magic(4)                     %输入 4 阶魔方矩阵
X =

    16     2     3    13
```

```
     5    11    10     8
     9     7     6    12
     4    14    15     1
≫y1 = diff(X)                    %一阶差分向量
y1 =
   -11     9     7    -5
     4    -4    -4     4
    -5     7     9   -11
≫y2 = diff(X,1,2)                %沿行求一阶差分向量
y2 =
   -14     1    10
     6    -1    -2
    -2    -1     6
    10     1   -14
```

【例 8-3】 已知$\boldsymbol{X}$ = [1 2 3 4 5 6]、$\boldsymbol{Y}$ = [1 4 9 16 25 36]，求向量 $\boldsymbol{Y}$，$\boldsymbol{X}$ 的近似微商之商 delta Y。

解：

```
≫X = 1:6                         %输入向量 X
X =
     1     2     3     4     5     6
≫Y = X.^2                        %输入向量 Y
Y =
     1     4     9    16    25    36
≫deltaY = diff(Y)./diff(X)       %近似微商
deltaY =
     3     5     7     9    11
```

【例 8-4】 已和时间向量 t = 1：10，距离向量$\boldsymbol{s}$ = [2 6 12 20 30 42 56 72 90 110]，求加速度向量 $\boldsymbol{a}$。

解：

```
≫t = 1:10;                       %输入时间向量
≫s = [2  6  12  20  30  42  56  72  90  110];   %输入距离向量
≫v = diff(s)./diff(t)            %计算速度向量
v =
     4     6     8    10    12    14    16    18    20
≫a = diff(v)./diff(t(1:9))       %计算加速度向量 a,为了与向量 v 同维,所以 t 的项数
                                  减 1
a =                              %加速度向量
     2     2     2     2     2     2     2     2
```

8.2 微分运算

向量的差分函数 diff 在符号数学工具箱里的是作为微分运算函数。当在命令窗口键入 doc symbolic/diff，即显示 diff 在符号数学工具箱中的用法。

diff(S) (8-4)

diff(S,'v') (8-5)

diff(S,'v',n) (8-6)

式中，S 为符号数学表达式，v 为 S 表达式中符号变量，n 为正整数。

式（8-4）是对 S 表达式中的缺省变量求微商，缺省变量由函数 findsym（寻找符号变量函数）确定。

式（8-5）是对 S 表达式中的符号变量 v 求微商。

式（8-6）是对 S 表达式中的符号变量 v 求 n 阶微商。

由式（8-4）~式（8-6）算出的为符号变量表达式，当变量由数值代入后，尚须用转换函数 eval 将微分表达式转换成数值。

【例 8-5】 已知函数 $y = x\sin x + 15$，求其微分表达式。并求 $x = \pi/4$ 时微商的值。

解:

```
≫syms x                          %设置符号变量 x
≫y = x* sin(x) + 15;             %输入符号数学表达式 y
≫diff(y)                         %求表达式 y 的微商
ans =
sin(x) + x* cos(x)
≫x = pi/4;
≫eval(ans)
ans =
    1.2625
```

【例 8-6】 求 $y = t^6$ 的 6 阶微商。

解:

```
≫syms t                          %设置符号变量 t
≫y = t^6;                        %输入符号变量表达式 y
≫diff(y,6)                       %取 6 阶微商
ans =
720
```

【例 8-7】 已知函数 $y = \exp(-t)\sin(xt)$，分别求 y 对 t 的偏微商和对 x 的偏微商。

解:

```
≫syms x t                        %设置符号变量 x、t
≫y = exp(-t)* sin(x* t);         %输入符号数学表达式
≫yt = diff(y,'t')                %求函数 y 对变量 t 的微商
yt =
-exp(-t)* sin(x* t) + exp(-t)* cos(x* t)* x
```

```
>>yt = simple(yt)                          %对 yt 进行化简
yt =
exp(-t)*(-sin(x*t)+cos(x*t)*x)
>>yx = diff(y,'x')                         %对变量 x 求导数
yx =
exp(-t)*cos(x*t)*t
```

【例 8-8】 已知参数方程 $x = a\cos^3 t$，$y = a\sin^3 t$，求导数 dy/dx。

解：

```
>>syms a t                                 %设置符号变量
>>x = a*cos(t)^3;                          %输入 x 符号数学表达式
>>y = a*sin(t)^3;                          %输入 y 符号数学表达式
>>x1 = diff(x)                             %计算 x1 = dx/dt
x1 =
-3*a*cos(t)^2*sin(t)
>>y1 = diff(y)                             %计算 y1 = dy/dt
y1 =
3*a*sin(t)^2*cos(t)
>>yx = x1/y1                               %计算 yx = dy/dx
yx =
-cos(t)/sin(t)
```

【例 8-9】 求隐函数 $x^2 + 2xy - y^2 = 2x$ 的导数。

解：

```
>>syms x y                                 %设置符号变量
>>z = x^2+2*x*y-y^2-2*x;                   %输入隐函数 z,z = 0
>>zx = diff(z,'x')                         %计算 z 对 x 的偏导数
zx =
2*x+2*y-2
>>zy = diff(z,'y')                         %计算 z 对 y 的偏导数
zy =
2*x-2*y
>>yx = -zx/zy                              % 考虑 z 的全微分为零,所以 y 对 x 的导数为
                                             -zx/zy
yx =
(-2*x-2*y+2)/(2*x-2*y)
>>yx = simple(yx)
yx =                                       %经化简后的隐函数导数
-(x+y-1)/(x-y)
```

【例 8-10】 求内摆线函数x^(2/3)+y^(2/3)=a^(2/3)的微商。

解：

```
≫syms x y a                              %设置符号变量
≫z = x^(2/3) + y^(2/3) - a^(2/3);        %输入内摆线函数
≫zx = diff(z,'x')                        %计算对 x 的偏微分
zx =
2/3/x^(1/3)
≫zy = diff(z,'y')                        %计算对 y 的偏微分
zy  =
2/3/y^(1/3)
≫yx = -zx/zy                             %考虑对 z 的全微分为零,则 y 对 x 的导数 yx 为
                                          -zx/zy
yx  =
-1/x^(1/3)*y^(1/3)
```

【例 8-11】 求 $y = \sin x\sin(2x)\sin(3x)$的 10 阶导数。

解:

```
≫syms x                                  %设符号变量
≫y = sin(x)*sin(2*x)*sin(3*x);           %输入函数 y
≫y10 = diff(y,10)                        %求 y 的 10 阶导数
y10 =
-15378944*sin(x)*sin(2*x)*sin(3*x) + 14854144*cos(x)*cos(2*x)*sin(3*x) +
14854656*cos(x)*sin(2*x)*cos(3*x) + 15378432*sin(x)*cos(2*x)*cos(3*x)
≫y10 = simple(y10)                       %化简
y10  =
-262144*sin(4*x) - 256*sin(2*x) + 15116544*sin(6*x)
≫factor(262144)                          %系数分解
ans =
  Columns 1 through 17
    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2    2
2
  Column 18
    2
≫factor(15116544)                        %系数分解
ans =
  Columns 1 through 17
    2    2    2    2    2    2    2    2    3    3    3    3    3    3    3    3
3
  Column 18
    3
≫y10 = -2^8*sin(2*x) - 2^18*sin(4*x) + 2^8*3^10*sin(6*x)    %y 的 10 阶导数
```

【例 8-12】 已知$y = x^2/(1-x)$，求 y 的 8 阶导数。

解：

```
≫syms x                               %设置符号变量
≫y = x^2/(1 - x);                     %输入函数 y
≫y8 = diff(y,8)                       %求 8 阶导数
y8 =
40320/(1 - x)^7 + 80640* x/(1 - x)^8 + 40320* x^2/(1 - x)^9
≫y8 = simple(y8)                      %化简
y8 =
-40320/(-1 + x)^9
≫for i = 1:15                         %检查 40320 是否是阶乘函数,factorial 为阶乘函
                                      数
if factorial(i) == 40320              %假若 40320 是阶乘函数,则中断循环
break
end
end,i
i  =                                  %运行结果,40320 是 8 的阶乘
    8
≫y8 = factorial(8)./(1 - x)^9         %8 阶导数可写成阶乘形式
```

【例 8-13】 已知 $y = \exp(t)(\sin t + \cos t)$，$t = \log(x)$，求 dy/dx。

解：

```
≫syms x t                             %设置符号变量
≫y = exp(t)* (sin(t) + cos(t));       %输入函数 y
≫y1 = diff(y)                         %取 y 微商
y1 =
exp(t)* (sin(t) + cos(t)) + exp(t)* (cos(t) - sin(t))
≫t = log(x);                          %输入函数 t
≫t1 = diff(t)                         %取 t 微商
t1 =
1/x
≫yx = y1* t1                          %dy/dx = dy/dt* dt/dx
yx =
(exp(t)* (sin(t) + cos(t)) + exp(t)* (cos(t) - sin(t)))/x
≫yx = simple(yx)
yx =
2* exp(t)* cos(t)/x
```

8.3 不定积分与定积分计算

不定积分是微分的逆运算，MATLAB 提供积分运算函数 int，它的书写格式如下：

R = int(S)

R = int(S,v)

R = int(S,a,b)

R = int(S,v,a,b)

式中，S 为被积函数的符号数学表达式，v 为指定的积分变量。R 为不定积分表达式。a、b 指定积分的下限和上限，它可以是符号变量或双精度的标量。

R = int(S)，返回 S 的不定积分，积分变量由 findsym 函数确定。

R = int(S,v)，返回 S 的不定积分，积分变量由 v 指定，v 指定的变量必须是符号变量。

R = int(S,a,b)和 R = int(S,v,a,b)则返回定积分计算，积分限由 a 到 b。

对于多重积分，可以对被积函数采用累次单变量积分来实现。若 S 是 x、y 的函数，积分限分别为[xa,xb]、[ya,yb]，则被积函数的二重积分为

R = int(int(S,x,xa,xb),y,ya,yb)

该式先对自变量 x 求积分，积分后的表达式含有自变量 y，随后再对变量 y 求积分。

【例 8-14】 已知 $S = 1/(1+x^2)$ 求积分表达式 R。

解：

```
≫syms x                              %设置符号变量
≫y = 1/(1 + x^2);                    %输入被积函数
≫R = int(y)                          %计算不定积分
R =
atan(x)
```

【例 8-15】 已知三角函数 $y = \cos x\cos(2x)\cos(3x)$，求积分表达式 R_1。

解：

```
≫syms x                              %设置符号变量
≫y = cos(x)*cos(2*x)*cos(3*x);       %输入被积函数
≫R1 = int(y)                         %计算不定积分
R1 =
1/8*sin(2*x) + 1/16*sin(4*x) + 1/24*sin(6*x) + 1/4*x
```

【例 8-16】 已知带参变量 a、b 的被积表达式 $y = \exp(ax)\sin(bx)$，求积分表达式 R_1。

解：

```
≫syms a b x                          %设置符号变量
≫y = exp(a*x)*sin(b*x);              %输入被积表达式
≫R = int(y)                          %计算不定积分
R =
-b/(a^2 + b^2)*exp(a*x)*cos(b*x) + a/(a^2 + b^2)*exp(a*x)*sin(b*x)
≫R1 = simple(R)                      %使用化简函数,化简被积表达式
R1 =
exp(a*x)*(-b*cos(b*x) + a*sin(b*x))/(a^2 + b^2)
```

【例 8-17】 已知 $y = \sqrt{1-\sin(2x)}$，求积分限为 $[0,\pi/2]$ 时的定积分。

解：

```
≫syms x                              %设符号变量
```

```
≫y = (1 - sin(2* x))^(1/2);                    %输入被积函数
≫R = int(y,0,pi/2)                             %计算定积分
R =
-2 + 2* 2^(1/2)
≫eval(R)                                        %计算定积分之值
ans =
    0.8284
```

【例 8-18】 已知椭圆的参数方程式为 $x = a\cos t$，$y = b\sin t$，求椭圆面积 A。

解：

椭圆的微分面积 $\mathrm{d}A = y\mathrm{d}x = -ab\sin^2 x$，被积函数 $S = -ab*\sin x$，在命令窗口输入如下程序：

```
≫syms a b t                                     %设置符号变量
≫S = -a* b* sin(t)^2;                           %输入被积表达式
≫A = int(S,2* pi,0)                             %计算椭圆面积为πab
A =
pi* a* b
```

【例 8-19】 已知抛物线椭圆函数 $Z = 10 - (X/2)^2 - (Y/3)^2$，求在区间 $X = [-4\ 4]$，$Y = [-4\ 4]$，曲面 Z 与 x、y 平面所围体积。

解：

```
≫syms x y                                                      %设置符号变量
≫ V = int(int(10 - x.^2/4 - y.^2/9,x,-4,4),y,-4,4)   % 用符号变量计算二重积分,先对变量
                                                                x求积,后对y变量求积
V =
13952/27
≫eval(V)                                                       %将字符串数字转换成数值
ans =
  516.7407
```

8.4 数值积分

由于通过符号数学表达式能解出的积分非常有限。所以有时还不得不使用数值积分，来代替符号数学积分。在 MATLAB 中，数值积分的函数有梯形积分法 trapz 和递推的辛普森积法函数 quad。梯形积分法使用简单，方法直观，但精度较差。递推的辛普森积分法，则精度较高。trapz 和 quad 是用于单变量积分。对于二重积分和三重积分则分别使用函数 dblquad 和 triplequad。下面分别于以叙述。

1. 梯形积分法

梯形积分法的书写格式如下：

$$z = \mathrm{trapz}(Y) \tag{8-7}$$

$$z = \mathrm{trapz}(X, Y) \tag{8-8}$$

式中，X 为输入向量，Y 为被积函数的输出向量。z 为积分值。对于式 (8-7) 是考虑步长为

单位长度的情况。若步长为 delta，则积分值 z 尚须乘以 delta。

【例 8-20】 已知 $\boldsymbol{Y}=[0\ 1\ 4\ 9\ 16\ 25\ 36]$，步长为 1，求梯形积分。

解：

```
≫Y = [0 1 4 9 16 25 36];                %输入向量 Y
≫z = trapz(Y)                           %用梯形积分法求向量 Y 的积分
z =
    73
≫cumtrapz(Y)                            %使用累加求梯形积分
ans =
    0    0.5000    3.0000    9.5000    22.0000    42.5000    73.0000
```

【例 8-21】 求 $y=\sin x$ 的均方根值。即 $z=\sqrt{1/\pi(\int_0^{pi}\sin^2 x\,\mathrm{d}x)}$，在 $[0,\pi]$ 区间。

解：

```
≫x = 0:pi/100:pi;                       %输入向量 x,步长为 pi/100
≫y1 = sin(x).^2;                        %计算正弦波的平方向量 y1
≫delta = pi/100;                        %输入步长 delta
≫z = (1/pi* delta* trapz(y1))^(1/2)     %计算方均根值,精确值应为 1/2* sqrt(2)
z =
    0.7071
```

【例 8-22】 已知椭圆抛物面函数 $Z=6-(X/2)^2-(Y/3)^2$，求在区间 $X=[-4,4]$，$Y=[-4,4]$，曲面 Z 与 x、y 平面所围体积。显示其表面图。

解：

```
≫syms x y                                        %设置符号变量
≫V = int(int(6 - x.^2/4 - y.^2/9,y, -4,4),x, -4,4)  % 用符号变量计算二重积分,先对变量 y
                                                    求积,后对 x 变量求积
V =
7040/27
≫eval(V)                                         %将字符串数字转换成数值
ans =
    260.7040
≫[X,Y] = meshgrid(-4:0.2:4);                     % 曲面 Z 与 X、Y 平面所围成的体积如图
                                                   8-1 所示
≫Z = 6 - (X/2).^2 - (Y/3).^2;
≫surfc(X,Y,Z)
```

2. 递推的辛普森积分法

递推的辛普森积分的书写格式为

```
q = quad(fun, a, b)
q = quad(fun, a, b, tol)
q = quad(fun, a, b, tol, trace)
```

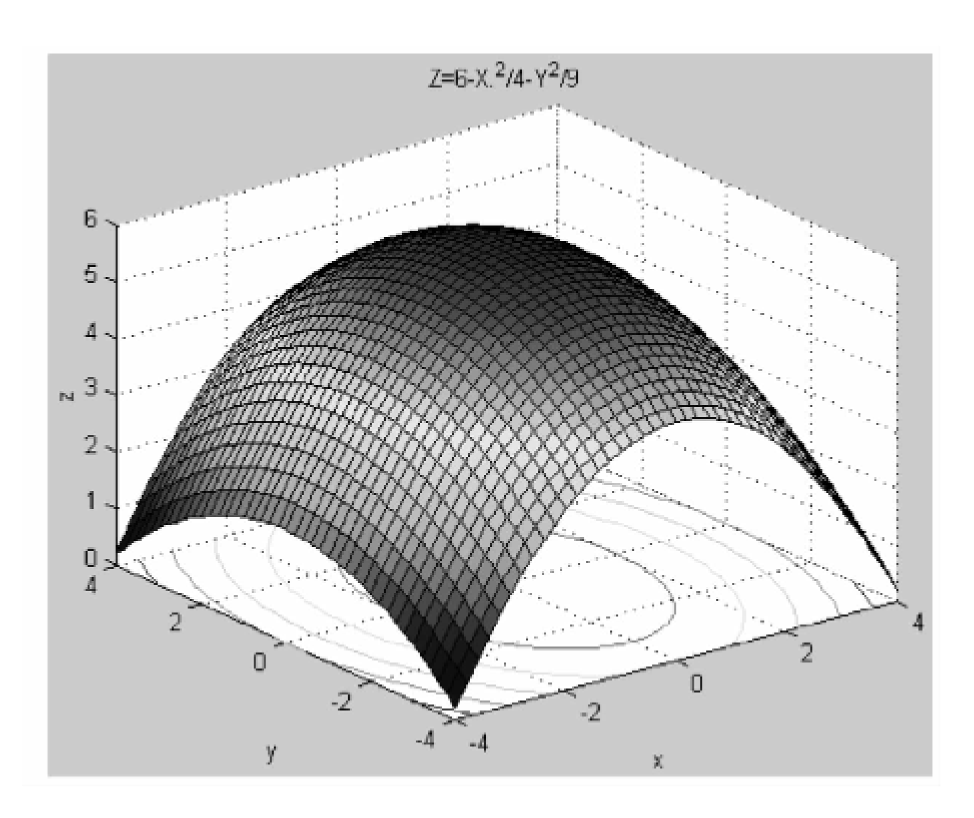


图 8-1　Z 的彩色表面图与 X、Y 平面所围体积

q = quad(fun，a b，tol，trace，p1，p2…)

[q, fcnt] = quadl(fun，a b…)

式中，fun 为被积函数和表达式，a、b 为积分的下限和上限。tol 为积分允许误差。Trace 设置为 1 则显示递推过程数据表，取 0 则不显示。p1，p2…用来对函数 fun 直接传递参数。fcnt 用来显示辛普森积分法，函数估值的次数。在允许误差缺省的情况下的计算精度为 1e-6。对于使用积分函数 quadl，则精度更高。

对于函数 fun 可以用下列 3 种方法来表示。

(1) 用字符串数学表达式，适用于单变量函数。

(2) 用内联函数 inline

(3) 用函数句柄，@myfun、myfun 是由用户自行编写的函数 M 文件，函数名也可任意制定。

3. 数值二重积分

数值二重积分的数学表达式为

$$V = \iint f(x, y)\,\mathrm{d}x\,\mathrm{d}y$$

式中，f(x, y)为被积函数，x 为内积分变量，y 为外积分变量。[xmin, xmax]、[ymin, ymax]分别为积分变量的上、下限。

二重积分的书写格式为

q = dblquad(fun, xmin, xmax, ymin, ymax, tol)

式中，fun 为被积函数，tol 为允许误差。

【例 8-23】　已知被积函数为 $y = \sin x/(x^3 + 3x + 5)$，求在[0, π]的梯形积分和辛普森积分。

解：

```
>>x = 0:pi/20:pi;                              %输入自变数向量
>>y = sin(x)./(x.^3 + 3* x + 5);               %计算被积函数向量
>>z = trapz(x,y)                               %采用梯形积分法
z =
    0.1616
>>Q = quad('sin(x)./(x.^3 + 3* x + 5)',0,pi)   %采用辛普森积分法
Q =
    0.1620
>>[Q1,fcnt] = quadl('sin(x)./(x.^3 + 3* x + 5)',0,pi)  %采用高精度算法
Q1 =
    0.1620
fcnt =
    48
```

【例 8-24】 已知椭圆的参数方程式 $x=4\cos t$，$y=3\sin t$，分别用梯形积分法和辛普森积分法求椭圆的周长。

解：

$$\text{周长的微分 } \mathrm{d}s = \sqrt{\mathrm{d}x^2+\mathrm{d}y^2} = \sqrt{16\sin^2 t+9\cos^2 t}\,\mathrm{d}t$$

所以被积函数 $S=\sqrt{16\sin^2 t+9\cos^2 t}$，为此编制函数 M 文件如下：

文件名为 func.m

```
%ellipse perimeter
function y = func(t)                              %函数定义项
y = (16* sin(t).^2 + 9* cos(t).^2).^(1/2);        %函数体,函数文件结束
```

在命令窗口输入如下程序：

```
>>t = 0:pi/50:2* pi;              %设置参变量向量
>>y = func(t);                    %计算周长的被积函数向量
>>z = trapz(t,y)                  %用梯形法,计算周长
z =
    22.1035
>>Q = quad(@func,0,2* pi)         % 用辛普森法计算周长,采用函数句柄,结果与梯形法相同
Q =
    22.1035
```

【例 8-25】 由【例 8-14】可知 $\int_0^1 \frac{\mathrm{d}x}{1+x^2}=\pi/4$，用数值积分计算 π 的值，精确到期 1e−6。

解：

```
>>format long                                    %设置15位显示格式
>>fun = inline('1./(1 + x.^2)')                  %设置内联函数
fun =
     Inline function:
     fun(x) = 1./(1 + x.^2)
>>s = quadl(fun,0,1,1e-8)                        %用数值积分计算面积,精度为1e-8
s =
     0.78539816339745
>>pia = 4* s                                     %设置π的近似值为pia(与MATLAB
                                                  的专用圆周率常数pi相区别)
pia =
     3.14159265358981
>>pi                                             %π的精确值,误差为6.4e-14
ans =
     3.14159265358979
```

【例 8-26】 已知三维曲面函数 $z = y\sin x + x\cos y$，变量 x 的积分限为 $[0, \pi/2]$，变量 y 的积分限为 $[0, \pi]$，求曲面 z 与 x、y 平面所围体积。

解：

```
>>fun = inline('y.* sin(x) + x.* cos(y)')        %将题设函数建立为内联函数
fun =
     Inline function:
     fun(x,y) = y.* sin(x) + x.* cos(y)
>>V = dblquad(fun,0,pi/2,0,pi)                   %计算二重积分
V =
     4.9348
```

4. 三重数值积分

三重数值积分的书写格式为

q = triplequad(fun,xmin,xmax,ymin,ymax,zmin,zmax,tol)

式中，fun 为被积函数表达式，xmin、xmax、ymin、ymax、zmin、zmax 为变量的积分的上、下限。Tol 为计算容许误差。

【例 8-27】 已知函数 $w = y\sin x + z\cos x$，变量 x 的积分限为 $[0, \pi]$，变量 y 的积分限为 $[0, 1]$，z 的积分限为 $[-1, 1]$，求数值积分。

解：

```
fun1 = inline('y.* sin(x) + z.* cos(x)')         %创建内联函数fun1
fun1 =
     Inline function:
     fun1(x,y,z) = y.* sin(x) + z.* cos(x)
>>V = triplequad(fun1,0,pi,0,1,-1,1)             计算三重积分
V =
```

2.0000

8.5 极限的计算

微积分中的一个重要内容是极限的计算。极限有数列极限与函数极限之分。对于函数的极限是这样定义的。对于任意给定一个正数 ε，必存在正数 δ，当

$0<|x-x_0|<\delta$ 时

$|f(x)-A|<\varepsilon$ 则称 $x \to x_0$ 时，$f(x)$ 的极限等于 A，写作

$\text{limit } f(x)=A$

在 MATLAB 中极限运算由函数 limit 来实现。其书写格式为

limit(F,x,a)——它取符号表达式 F，在 x→x0 时的极限；

limit(F,a)——由 findsym(F)确定的变量作为自变量；

limit(F)——用 a=0 作为极限点；

limit(F,x,a,'right')或 limit(F,x,a,'left')——规定取极限的方向。

【例 8-28】 计算 $m/(1-x^m)-n/(1-x^n)$ 在 $x\to 1$ 时的极限。

解：

```
>>syms m n x                          %设置符号变量
>>F = m/(1 - x^m) - n/(1 - x^n);      %输入数学表达式
>>limit(F,x,1)                        %取极限
ans =
1/2* m - 1/2* n
```

【例 8-29】 计算 $\dfrac{\sqrt{x+\sqrt{x+\sqrt{x}}}}{\sqrt{x+1}}$，在 $x\to$inf 时的极限。

解：

```
>>syms x
>>F = (x + (x + (x)^(1/2))^(1/2))^(1/2)/(x + 1)^(1/2);
>>limit(F,x,inf)
    ans =
        1
```

【例 8-30】 计算 $\dfrac{\sqrt[n]{1+x}-1}{x}$，求 $x\to 0$ 时的极限。

解：

```
>>syms x n
>>F = ((1 + x)^(1/n) - 1)/x;
>>limit(F)
ans =
1/n
```

【例 8-31】 计算 $\{(x+a/n)+(x+2a/n)+\cdots+[x+(n-1)a/n]\}/n$，当 $n\to$inf 时的极限。

解：

```
≫syms a k n x                           %设置符号变量,k=0,1,2,…,(n-2)
≫S=x+(k+1)*a/n;                         %级数求和表达式
≫s=simple(symsum(S,k))                  %求级数和并化简,其中包括括号内的内容
s=
1/2*k*(2*x*n+a+a*k)/n
≫s=subs(s,k,n-2)                        %用n-2替换变量k
s=
1/2*(n-2)*(2*x*n+a+a*(n-2))/n
≫limit(s/n,n,inf)                       %计算极限
ans=
x+1/2*a
```

【例 8-32】 计算 $\cos 2x/\sqrt{1-\sin 2x}$，在 $x=\pi/4$ 时的左、右极限。

解：

```
≫syms x                                 %设置符号变量
≫y=cos(2*x)/sqrt(1-sin(2*x));           %输入计算极限表达式
≫limit(y,x,pi/4,'left')                 %计算左极限
ans=
2^(1/2)
≫limit(y,x,pi/4,'right')                %计算右极限
ans=
-2^(1/2)
```

由于在 x=pi/4 时，函数的左、右极限不等，故函数在此点是不连续的。

8.6 常微分方程的符号解

在科学研究和工程设计中，经常会碰到微分方程问题，例如单摆运动，物体的冷却过程，电阻、电感和电容的串联谐振，汽车中避振弹簧的设计，电杆上悬挂导线所形成的悬链线，导弹的跟踪轨迹等，都是微分方程的例子。

微分方程式是包含有未知函数及其导数的方程。若未知函数中仅含一个自变量，则这种微分方程称为常微分方程式。若未知函数中含有一个以上的自变量，则称为偏微分方程式。若微分方程式中的各阶导数和未知函数都是一次的，这种方程式称为线性微分方程式。若微分方程式中的各价导数的系数均为常数，则称为常系数线性微分方程式。

微分方程式的阶次，是以方程式中最高导数的阶次来计量。微分方程中未知函数及其导数有关自变量在同一点的辅助条件构成初值问题或称为初始条件。若给出自变量一个以上点的条件则称作边界条件。

MATLAB 中解微分方程式的函数为 dsolve，它是属于符号数学工具箱的函数。它的书写格式为

```
R=dsolve('eq1,eq2…','cond1,cond2,…'v')
```

式中，eq1，eq2，…为一般的微分方程式，cond1、cond2，…为初始条件，v 为相关的自变量。v 的默认自变量为 t。在 MATLAB 中是用大写 D 作为相关于自变量的微分标记，即

d/dx。2 阶微分 d2/dx2 用D2 表示。任何跟在标记 D 后的应变量，即执行微分操作。初始条件用y(a) = b,或者 D(a) = b 表示，放置在以初始条件栏里。dsolve 最多接受 12 条语句，即允许 eq1，eq2，…，eq12。

【例 8-33】 求解 2 阶线性微分方程式 $d^2y/d^2x+4y=\sin x$。

解：

```
≫dsolve('D2y + 4* y = sin(x)','x')        %解微分方程式,自变量设为 x
ans =                                      %微分方程式的解,前两部分为补解,后一部分为
                                             特解,合起来为通解。答案中的 C1、C2 为积分常
                                             数,由初始条件来确定
sin(2* x)* C2 + cos(2* x)* C1 + 1/3* sin(x)
```

【例 8-34】 求解微分方程式 $dy/dx=-x/y$。

解：

```
≫dsolve('Dy = - x/y','x')
ans =                                      %解答,负的解答代入后不适合,所以正解为 x^2 +
                                             y^2 = C1
[(- x^2 + C1)^(1/2)]
[- (- x^2 + C1)^(1/2)]
```

【例 8-35】 在一个串联的 R、L 电路，施加交流电压 $u\sin(wt)$，求回路电流。考虑初始条件为 $I(0)=0$，$dI(0)/dt=0$。

解：

设电流为 I，时间常数 $T=L/R$，由电路方程式可得

$$LdI/dt+IR=u\sin wt$$

写成标准形式为

$$dI/dt+I/T=u/L\sin wt=I_0\sin wt$$

在命令窗口输入如下程序：

```
≫clear                                               %清内存
≫dsolve('DI + I/T = I0* sin(w* t)','I(0) = 0,DI(0) = 0')  %解微分方程式,输入初始条件
ans =
exp(- 1/T* t)/(1 + w^2* T^2)* I0* w* T^2 - I0* T* (w* T* cos(w* t) - sin(w* t))/(1 + w^2* T
^2)
≫I = simple(ans)                                     %化简
I =
I0* T* (exp(- 1/T* t)* w* T - w* T* cos(w* t) + sin(w* t))/(1 + w^2* T^2)
```

用手工写成通常表达形式为

I = u/R* wt/(1 + wT)^(1/2)* (exp(- t/T) + sin(wt - zeta))

式中，zeta = atan(wT)

【例 8-36】 追线是微分方程的经典问题，它有多种叙述方法，如狗追逐兔子，导弹攻击飞机等。本例则以导弹为例。设在 x 轴正向 b 处有架敌机，并行于 y 轴沿正向等速飞行。在坐标原点处有地空导弹发射场发射导弹，导弹运行轨迹的切线方向随时对准敌机。求导弹

的运行轨迹。若敌机的航速为 $v=0.2\text{km/s}$，导弹的线速为 nv，问几秒能击中目标，击中点的纵坐标。参看图 8-2 导弹运行轨迹。

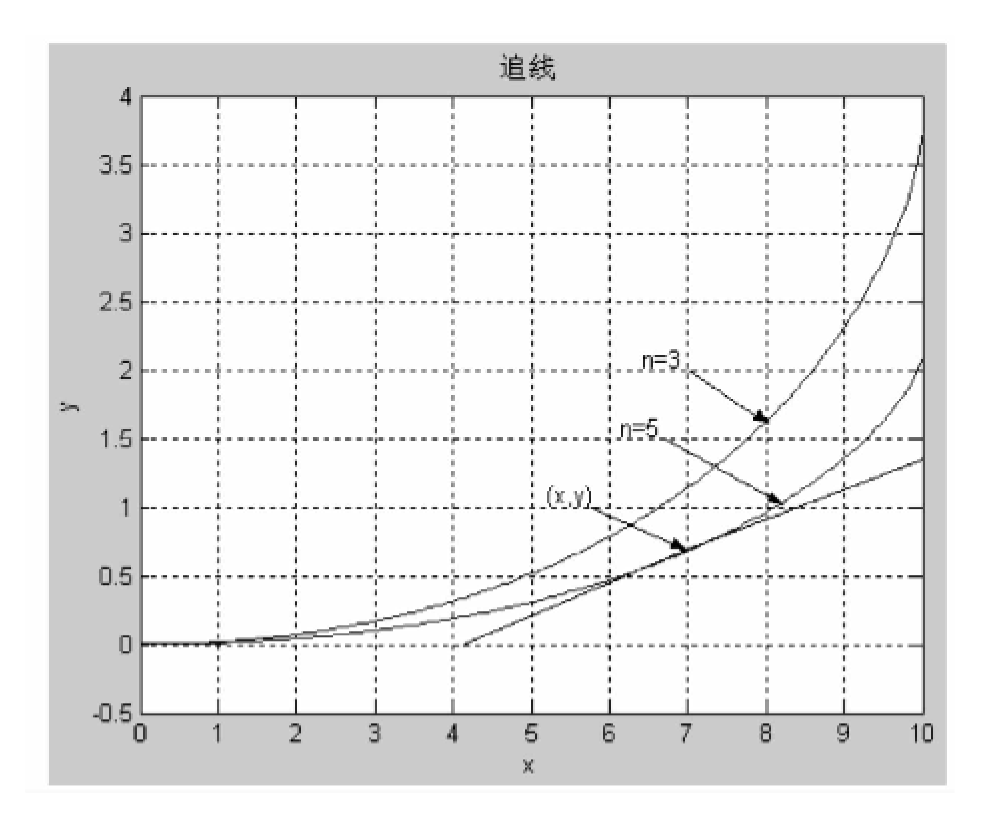


图 8-2 导弹运行轨迹

解：

追线的微分方程式 $\qquad dy/dx=(vt-y)/(b-x)$ (8-9)

弧线长 $\qquad \text{int}(\text{sqrt}(1+(dy/dx)^2),0,x)=nvt$ (8-10)

%int 为求积函数

式(8-10)代入式(8-9)得 $\qquad (b-x)dy/dx+y=i/n\,\text{int}(\text{sqrt}(1+(dy/dx)^2),0,x)$ (8-11)

式两边对式 x 取微分得 $(b-x)d^2y/d^2x=1/n((1+(dy/dx)^2)^{1/2}$ %d2y/d2x 为二次微商

令 $p=d/dx$ 得 $\qquad dp/dx=(b-x)/n\sqrt{1+p^2}$ (8-12)

分离变量，对式(8-12)改写为 $\qquad dp/(1+p^2)^{1/2}=dx/n/(b-x)$ (8-13)

对式(8-13)两边进行积分得 $\qquad \lg(p+(1+p^2)^{1/2})=\lg(C(b-x)^{-1/n})$ (8-14)

考虑初始条件 $x=0$、$p=0$，得 $\qquad C=b^{1/n}$ (8-15)

将 C 代入，得 $\qquad p+(1+p^2)^{1/2}=b^{1/n}(b-x)^{-1/n}$ (8-16)

以上是对求解追线的前期工作，以下是在命令窗口输入的程序：

```
≫syms b n x y t p                                          %设置符号变量
≫ solve(p+(1+p^2)^(1/2)-b^(1/n)*(b-x)^(-1/n),'p')          %解代数方程
ans =
1/2*(-1+(b^(1/n))^2*((b-x)^(-1/n))^2)/(b^(1/n))/((b-x)^(-1/n))
≫p = simple(ans)                                           %化简
```

```
p =
-1/2*(b-x)^(1/n)*b^(-1/n)+1/2*b^(1/n)*(b-x)^(-1/n)
>>dsolve('Dy=-1/2*(b-x)^(1/n)*b^(-1/n)+1/2*b^(1/n)*(b-x)^(-1/n)','y(0)=0','x')
ans =                                           %解微分方程式，输入初始条件
1/2*b^(-1/n)*(b-x)^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x)^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2)
>>y=simple(ans)                                 %化简
y =
1/2*b^(-1/n)*(b-x)^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x)^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2)
>>n=3;                                          %设速度比 n=3、b=10km
>>b=10;
>>x=0:0.05:10;
>>y1=1/2*b^(-1/n)*(b-x).^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x).^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2);
>>plot(x,y1)                                    % 绘制导弹轨迹，如图 8-2 所示
>>hold on                                       %图形保持
>>n=5;                                          %设速度比
>>y2=1/2*b^(-1/n)*(b-x).^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x).^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2);
>>plot(x,y2),grid on                            % 绘制导弹轨迹，如图 8-2 所示
>>x1=10;
>>y21=1/2*b^(-1/n)*(b-x1).^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x1).^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2)
y21 =                                           %计算 n=5 时的击中距离
    2.0833
>>n=3;
>>y11=1/2*b^(-1/n)*(b-x1).^(1/n+1)/(1/n+1)-1/2*b^(1/n)*(b-x1).^(-1/n+1)/(-1/n+1)+b*n/(-1+n^2)
y11 =                                           %计算 n=3 时的击中距离
    3.7500
>>v=0.2                                         %设置 v=0.2km/s
v =
    0.2000
>>t1=3.75/0.2                                   % n=3 时的击中时间，单
```

位为秒

```
t1 =
    18.7500
≫t2 = 2.0833/0.2                          % n = 5 时的击中时间，单位为秒
t2 =
    10.4165
```

8.7 平面曲线族的包络线

含有一个可变参数 a 的曲线族f(x,y,a) = 0，如果它存在有包络线，则必定满足（参阅数学手册）

$$f(x, y, a) = 0, \mathrm{d}f/\mathrm{d}a = 0$$

由包络线的包络方程，可以了解曲线族的变化范围。利用 MATLAB 求微商的函数 diff，解代数方程式的函数 solve 和绘制线性图的函数 plot，那么包络线方程、包络线和曲线族图形，很容易得到解答。

【例 8-37】 考虑一根刚性的直线，长度为 c，在 x、y 平面正向坐标轴上连续滑动。求滑动直线的包络线。

解：

```
≫syms x y a c theta                      %设置符号变量,其中 a 为滑动直线在x轴上的截距,theta 为滑动直线与 x 轴的夹角
≫f = x/a + y/(c^2 - a^2)^(1/2) - 1;      %输入定长滑动线段方程式
eq = diff(f,'a')                         %对可变参数 a 取微商,并使之为零
eq =
- x/a^2 + y/(c^2 - a^2)^(3/2)*a
≫[x,y] = solve(f,eq)                     % 解代数方程式 f、eq,即得包络方程式的解
x =
a^3/c^2
y =
- (c^2 - a^2)^(1/2)*(- c^2 + a^2)/c^2
%令 cos(theta) = a/c,sin(theta) = (c^2 - a^2)^(1/2)/c,将其代入 x、y 消去 a、b、c,得
≫x = c*cos(theta)^3;                     %包络方程式的参数方程式形式
≫y = c*sin(theta)^3;
≫x^(2/3) + y^(2/3)                       % 包络方程式的化简形式为 x^(2/3) + y^(2/3) = c^(2/3)
ans =
(c*cos(theta)^3)^(2/3) + (c*sin(theta)^3)^(2/3)
≫ans1 = simple(ans)                      %化简结果
```

```
ans1 =
c^(2/3)*(cos(theta)^2 + sin(theta)^2)
>>ans1 = c^(2/3)
ans1 =
c^(2/3)
>>c = 5;                                    %设置滑动线段长
>>for a = 0.5:0.5:5                         %绘制截距 a 间断变化时的滑动直线图
        x = 0:0.05:5;
        y = (1 - x/a)*(c^2 - a^2)^(1/2);
        plot(x,y),hold on
    end
>> axis([0 5 -0.5 5.5])                     %坐标轴设置
>> theta = 0:pi/50:pi/2;                    %包络线参变量设置
        x1 = c*cos(theta)^3; y1 = c*sin(theta)^3;   %计算包络线坐标
        plot(x1,y1,'-k')                    %绘制包络线,如图 8-3 所示
        plot(,x1,y1,'or')                   %用圆点显示包络线的标记点
>>grid on                                   %添加栅格线
```

【例 8-38】 某大炮的炮弹的出口速度为 v，发射角为 a，若不计空气阻力，炮弹的运行轨迹为抛物线，求不同发射角 a 的情况下的抛物线的包络线。(设 $v = 100\text{m/s}$)，并求最大射程和最大射程相对应的最大高度。

解：

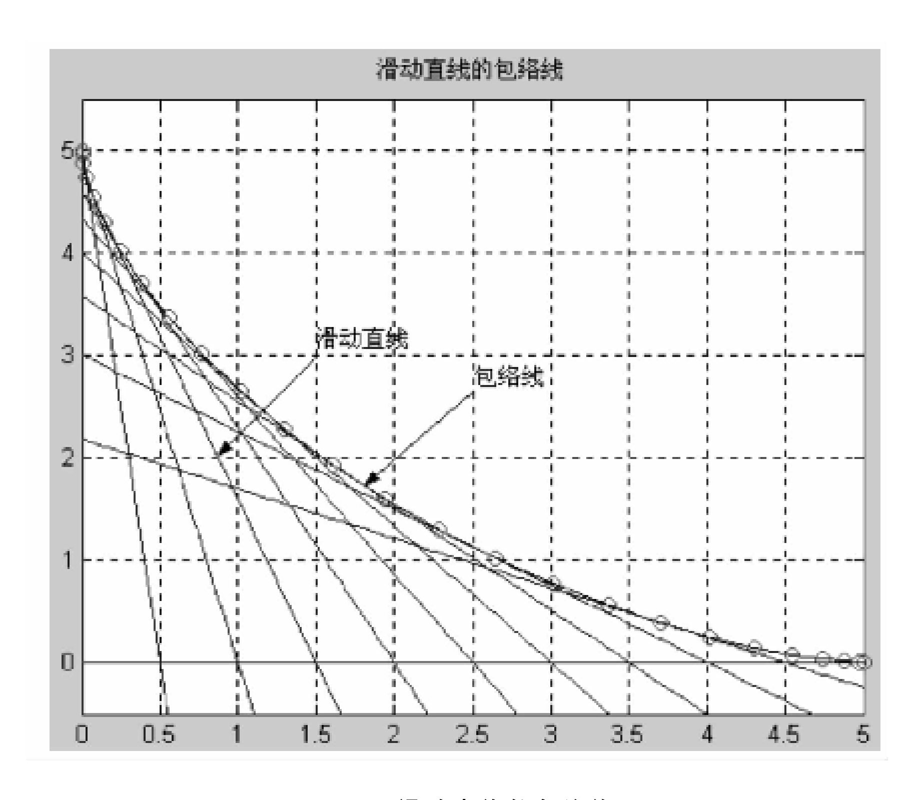


图 8-3　滑动直线的包络线

```
≫syms a x v g                                      %设置符号变量,g为重力加速度常数
≫y = x* tan(a) - g* x^2/2/v^2/cos(a)^2;            %输入抛物线方程式,该式由 x = vcos(a)t,
                                                     y = vsin(a)t - 1/2 gt^2 消去变量 t 而得
≫diff(y,'a')                                       %求 y 对 a 的微商
ans =
x* (1 + tan(a)^2) - g* x^2/v^2/cos(a)^3* sin(a)
≫ans1 = ans;
≫solve(ans1,'x')                                   %求解 x
ans =
[                                        0]
[(1 + tan(a)^2)* v^2* cos(a)^3/g/sin(a)]
≫x = ans(2)                                        %x 的非零解
x =
(1 + tan(a)^2)* v^2* cos(a)^3/g/sin(a)
≫x = simple(x)                                     %化简,得包络线横坐标的参数方程
x =
v^2* cos(a)/g/sin(a)
≫y = v^2/g* (1 - 1/2/sin(a)^2);                    %以 x 代入 y,得包络线纵坐标的参数方程
≫g = 9.8; v = 100;                                 %对变量 g , v 赋值
≫parabola_cover                                    %调用绘制包线程序,程序见下一页
≫hold on                                           %保存图形
≫parabola                                          %调用绘制抛物线程序见后,包络线及抛物
                                                     线图,如图 8-4 所示
≫y = x* tan(a) - g* x^2/2/v^2/cos(a)^2;            %为了计算最大射程,重写抛物线方程
≫solve(y)                                          %解方程,得射程
%ans =
[                              0]
[2* tan(a)* v^2* cos(a)^2/g]
≫x1 = simple(ans(2))                               %化简
x1 =
2* sin(a)* cos(a)* v^2/g
≫x1 = sin(2* a)* v^2/g;                            %射程表达式,显然在 a = π/4 时射程最大
≫x1max = v^2/g;                                    %最大射程表达式
≫diff(y)                                           %为求抛物线高度,先求对 y 的微商
ans =
tan(a) - g* x/v^2/cos(a)^2
≫solve(ans)                                        %求 y 微商为零的解
ans =
```

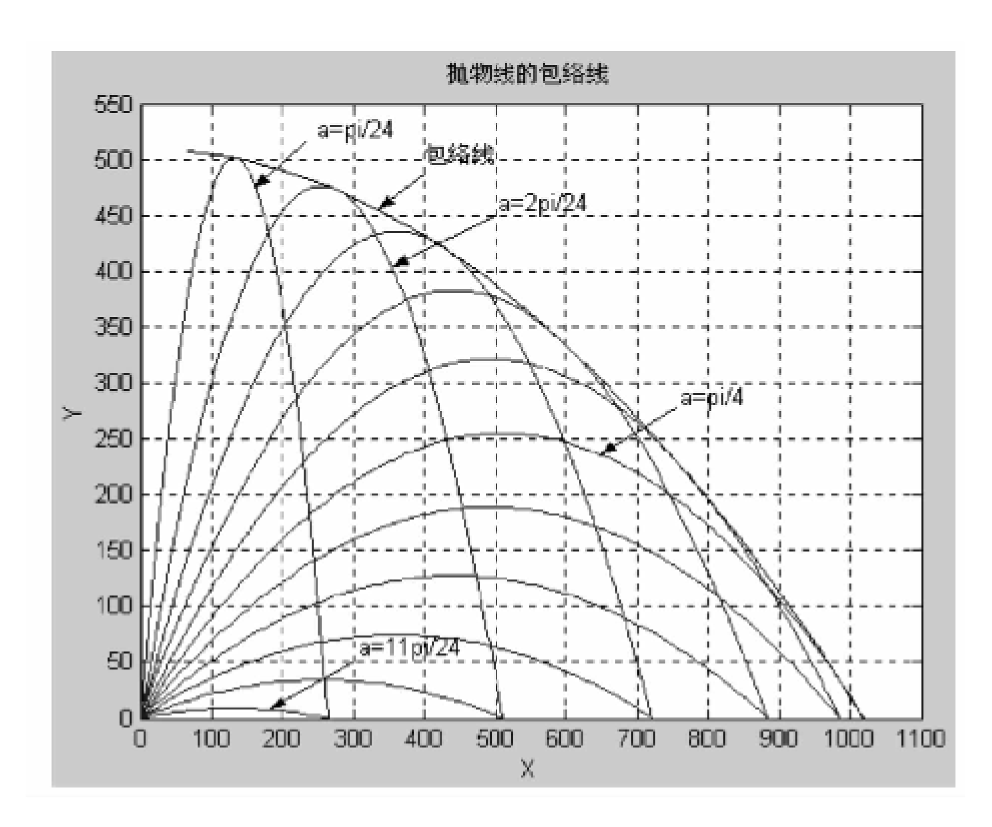


图 8-4 抛物线的包络线

```
tan(a)*v^2*cos(a)^2/g
>>x2 = ans;                              %解赋值给 x2,x2 为抛物线最大值时横坐标
>>ymax = subs(y,x2,x)                    %将 x2 代入,得最大高度 ymax
ymax =                                   %最大高度的符号表达式
1/2*tan(a)^2*v^2*cos(a)^2/g
>>a = pi/4;v = 100; g = 9.8;             %将符号变量用数字代入
>>x1max = v^2/g
x1max =                                  %最大射程
  1.0204e+003
>>ymax = 1/2*tan(a)^2*v^2*cos(a)^2/g
ymax =                                   %最大射程时的最大高度
  255.1020
```

绘制抛物线的包络线的程序如下,程序名为 parabola_cover

```
%This program for plot a line of parabola cover    %程序标题
a = pi/50:pi/50:24*pi/50;                          %设置发射角向量
    x = v^2/g*cos(a)./sin(a);                      %计算包络线
    y = v^2/g*(1-1/2./sin(a).^2);
    plot(x,y,'k-')                                 %绘制包络线,参阅图 8-4
axis([0,1100,0,550])                               %坐标轴设置
```

绘制发射角为 pi/12:pi/12:11*pi/12 抛物线族程序如下,程序名为 parabola

```
% This program for plot parabola group         %程序标题
for a = pi/24:pi/24:11* pi/24;                 %发射角变化范围
    t = 0:0.1:25;                              %设置时间向量
    x = 100* cos(a)* t;                        %计算抛物线的横坐标和纵坐标
    y = 100* sin(a)* t - 0.5* 9.8* t.^2;
    plot(x,y),hold on                          %绘制抛物线,参阅图 8-4
end                                            %发射角循环
axis([0,1100,0,550]),grid on                   %设置坐标,添加栅格线
```

8.8 常微分方程的数值解

常微分方程的解析解，由于求解的难度较高；有时解析解太复杂，难以使用；也有的根本上无法取得解析解，所以在工程上往往求助于数值解。所谓数值解，就是用加减乘除和函数求值等运算，在特定点求近似解的过程。求解常微分方程的数值解的方法有欧拉法，改进的欧拉法和龙格—库塔法。

(1) 欧拉法最为简单，它属于单步递推法，但精度也最差。若一阶微分方程为

$$y'_n = f(x_n, y_n)$$

则

$$y_{n+1} = y_n + hf(x_n, y_n)$$

而

$$y_0 = y(x_0)$$

式中，h 为步长。显然步长越小，误差也小。

(2) 改进的欧拉法是属于预测校正法，它是将 $[x_n,\ x_{n+1}]$ 左右两个点的斜率平均值作为 y_{n+1} 值的计算。即第一步计算预测值

$$py_{n+1} = y_n + hy'_n$$

第二步计算 $n+1$ 点处的斜率

$$py'_{n+1} = f(x_{n+1}, py_{n+1})$$

第三步计算校正值

$$y_{n+1} = y_n + \frac{h}{2}(py'_{n+1} + y'_n)$$

(3) 龙格—库塔法（Runge - Kutta Method）为

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$k_1 = hf(x_n, y_n)$$

$$k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})$$

$$k_3 = hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2})$$

$$k_4 = hf(x_n + h, y_n + k_3)$$

MATLAB 基本上采用龙格—库塔法来解常微分方程的数值解。解法函数有 ode23、ode45、ode113、ode23t、0de15s、ode23tb 等。书写格式为

[T,Y] = solver(odefun,tspan,y0)

$$[T, Y] = solver(odefun, tspan, y0, option)$$

式中，T、Y 为时间向量和对应输出向量，solver 为选择解法函数，如 ode23、ode45 等，odefun 为一阶微分方程组的函数名，若为高阶微分方程式，则必须将其化成一阶形式。tspan 为选择积分区间。y0 为选择初始条件向量。options 为选项，如选择计算精度、算法等。详见 odeset 函数（微分方程设置函数）。

【例 8-39】 已知一阶微分方程式 $dy/dx = 3y/(1+x)$，初始条件为 $y(0) = 1$，求 $\boldsymbol{x} = [0:0.1:1]$ 时的 $\boldsymbol{y}$ 向量。

解：

先建立微分方程式函数，在 M 文件编辑器中编制函数文件如下，程序名为 funode1

```
function dy = funode1(x,y)
dy = 3*y/(1+x);
```

在命令窗口输入如下程序：

```
>> [x1,y1] = ode23('funode1',[0:0.1:1],1);      %用 ode23 函数,求一阶微分方程式的数值解
>>x1'
ans =
  Columns 1 through 8
        0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000
  Columns 9 through 11
   0.8000    0.9000    1.0000
>> y1'
ans  =
  Columns 1 through 8
    1.0000    1.3308    1.7276    2.1963    2.7430    3.3736    4.0941    4.9106
  Columns 9 through 11
   5.8291    6.8554    7.9958
>>dsolve('Dy = 3*y/(1+x)','y(0) = 1','x')      %用符号解,解上述微分方程式
ans =
1 + 3*x + 3*x^2 + x^3
>>x = [0:0.1:1];
>> y = ans;
>>y = 1 + 3*x + 3*x.^2 + x.^3      %微分方程式的精确解
y =
  Columns 1 through 8
    1.0000    1.3310    1.7280    2.1970    2.7440    3.3750    4.0960    4.9130
  Columns 9 through 11
```

```
    5.8320    6.8590    8.0000
>>err = y - y1'                                    %精确解与数值解之差。小于 0.42%
err =
  Columns 1 through 8
       0    0.0002    0.0004    0.0007    0.0010    0.0014    0.0019
0.0024
  Columns 9 through 11
    0.0029    0.0036    0.0042
>>plotyy(x,y,x,err)                                % 绘制双纵坐标曲线,如图 8-5 所示,
                                                     右侧纵坐标为 y,左侧纵坐标为精确
                                                     解与数值解之差
```

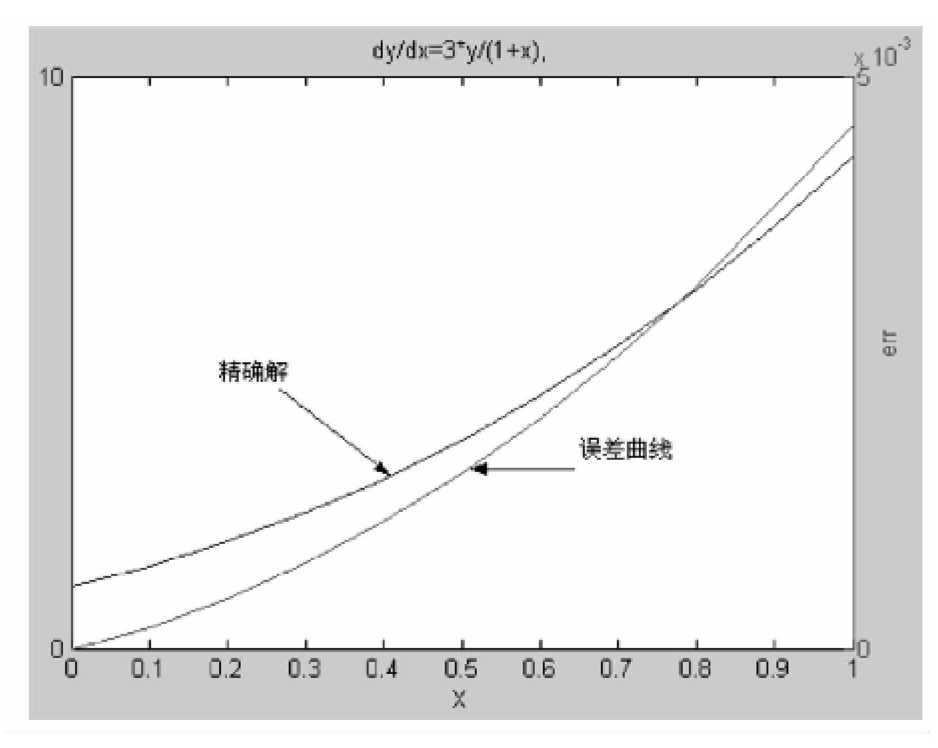


图 8-5　微分方程 $dy/dx = 3y/(1+x)$ 精确解与数值解的误差曲线

【例 8-40】　考虑一个刚体未受外力作用的微分方程为

$$y'(1) = y(2)y(3)$$
$$y'(2) = -y(1)y(3)$$
$$y'(3) = -0.51y(1)y(2)$$

已知初始条件为 $t=0$ 时，$y(0,1)=0, y(0,2)=1, y(0,3)=1$，考虑观察时间范围为[0，12]，求微分方程组的解。

解：

打开 M 文件编辑器，先建立微分方程组的函数 M 文件。函数名 funode2

```
function dydt = funode2(t,y)                                 %函数定义行
dydt = [y(2)*y(3); -y(1)*y(3); -0.51*y(1)*y(2)];   % 函数体,建立微分方程组的关系
                                                               向量
```

在命令窗口输入如下程序：

```
>>[t,y]=ode45(@funode2,[0,12],[0;1;1]);              % 使用 ode45 解微分方程组,tspan
                                                      =[0,12],初始条件 y0=[0;1;
                                                      1]
>> plot(t,y(:,1),'-k',t,y(:,2),':r',t,y(:,3),'-.g')  %绘制图形
>>hold on                                             %图形保持
>>plot(t,y(:,1),'or',t,y(:,2),'+g',t,y(:,3),'*k')    %添加标记点,图形如图 8-6 所示
```

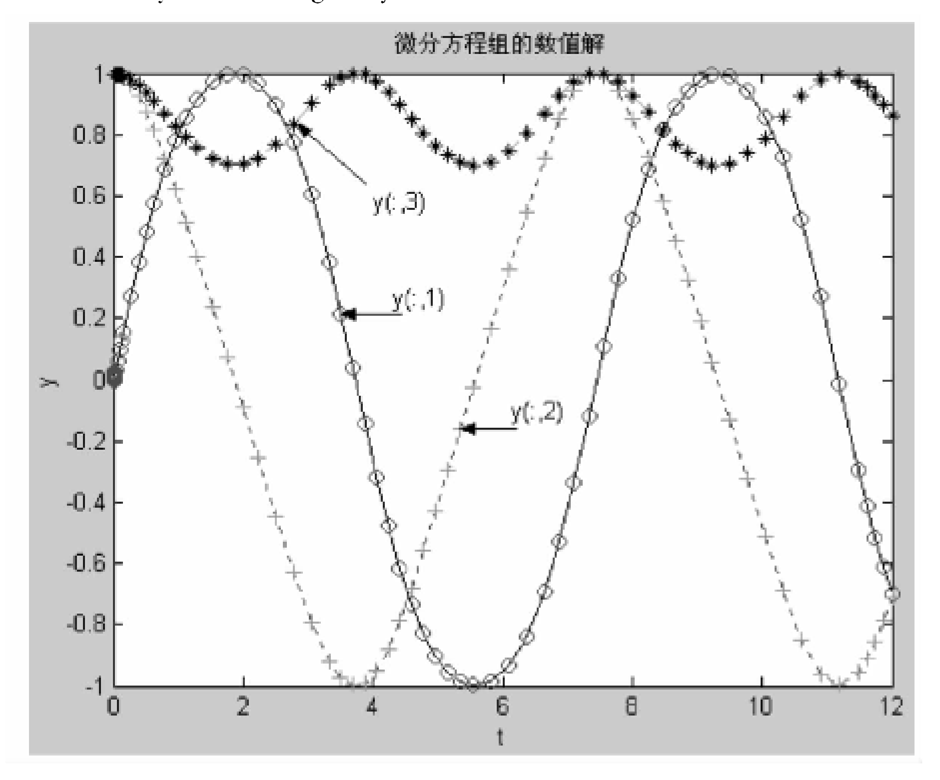


图 8-6 刚体不受外力时,微分方程组的解

```
>>tytable=[t,y];                                      %建立 t、y 数据表
>>tytable
tytable =                                             %输出数据
```

t	y(1)	y(2)	y(3)
0	0	1.0000	1.0000
0.0001	0.0001	1.0000	1.0000
0.0001	0.0001	1.0000	1.0000
0.0002	0.0002	1.0000	1.0000
0.0002	0.0002	1.0000	1.0000
0.0005	0.0005	1.0000	1.0000
0.0007	0.0007	1.0000	1.0000
0.0010	0.0010	1.0000	1.0000
0.0012	0.0012	1.0000	1.0000
0.0025	0.0025	1.0000	1.0000

0.0037	0.0037	1.0000	1.0000
0.0050	0.0050	1.0000	1.0000
0.0062	0.0062	1.0000	1.0000
0.0125	0.0125	0.9999	1.0000
0.0188	0.0188	0.9998	0.9999
0.0251	0.0251	0.9997	0.9998
0.0313	0.0313	0.9995	0.9997
0.0627	0.0627	0.9980	0.9990
0.0941	0.0939	0.9956	0.9977
0.1255	0.1250	0.9922	0.9960
0.1569	0.1560	0.9878	0.9938
0.2760	0.2709	0.9626	0.9811
0.3951	0.3803	0.9249	0.9624
0.5143	0.4824	0.8760	0.9388
0.6334	0.5758	0.8176	0.9115
0.7975	0.6888	0.7249	0.8707
0.9616	0.7829	0.6222	0.8291
1.1257	0.8583	0.5129	0.7900
1.2898	0.9162	0.4002	0.7560
1.5240	0.9713	0.2371	0.7201
1.7582	0.9971	0.0731	0.7019
1.9924	0.9956	-0.0909	0.7030
2.2267	0.9666	-0.2546	0.7232
2.5015	0.8954	-0.4458	0.7691
2.7764	0.7764	-0.6300	0.8322
3.0513	0.6052	-0.7953	0.9014
3.3262	0.3838	-0.9231	0.9616
3.5080	0.2146	-0.9764	0.9880
3.6898	0.0355	-0.9992	0.9996
3.8717	-0.1451	-0.9894	0.9947
4.0535	-0.3191	-0.9478	0.9739
4.2353	-0.4785	-0.8782	0.9400
4.4172	-0.6180	-0.7864	0.8976
4.5990	-0.7347	-0.6784	0.8515
4.7808	-0.8280	-0.5602	0.8063
4.9737	-0.9030	-0.4290	0.7642
5.1666	-0.9552	-0.2949	0.7311
5.3594	-0.9868	-0.1600	0.7093

5.5523	-0.9992	-0.0250	0.7003
5.8266	-0.9857	0.1667	0.7101
6.1008	-0.9331	0.3584	0.7454
6.3750	-0.8367	0.5467	0.8016
6.6493	-0.6911	0.7223	0.8696
6.8774	-0.5294	0.8482	0.9258
7.1056	-0.3351	0.9421	0.9710
7.3338	-0.1175	0.9929	0.9964
7.5619	0.1096	0.9939	0.9969
7.7901	0.3287	0.9443	0.9719
8.0182	0.5244	0.8517	0.9273
8.2464	0.6871	0.7269	0.8717
8.4745	0.8132	0.5817	0.8142
8.6643	0.8912	0.4534	0.7715
8.8541	0.9467	0.3217	0.7370
9.0439	0.9817	0.1889	0.7131
9.2337	0.9981	0.0561	0.7014
9.5047	0.9909	-0.1335	0.7066
9.7758	0.9459	-0.3232	0.7373
10.0468	0.8594	-0.5105	0.7894
10.3179	0.7257	-0.6876	0.8552
10.6096	0.5228	-0.8524	0.9281
10.9013	0.2695	-0.9631	0.9815
11.1930	-0.0118	-0.9990	0.9992
11.4847	-0.2936	-0.9540	0.9763
11.6136	-0.4098	-0.9102	0.9548
11.7424	-0.5169	-0.8539	0.9279
11.8712	-0.6135	-0.7874	0.8974
12.0000	-0.6987	-0.7128	0.8650

8.9 差分方程的求解

在科学研究和工程设计中，由于变量出现的形式不同，存在连续控制系统和离散控制系统。微分方程是用来解算连续量，而差分方程则用来解算离散量。由于数字控制技术的发展，差分方程的使用也逐渐普遍。其次在生物控制和经济分析也得到广泛的应用。差分方程最简单的例子是存款利息的计算。若存款本金为$y(0)$，年利率为 i，则 n 年后，同期的本息和为

$$y(n)=y(n-1)(1+i)=y(0)(1+i)^n$$

该式表示了第 n 年的本息和与第 $n-1$ 年本息和的递推关系，它属于一阶差分方程式。该式后面的等式则表示了差分方程式的解。

k 阶常系数线性差分方程式的标准形式为

$$a_k x(k) + a_{k-1} x(k-1) + a_{k-2} x(k-2) + \cdots + a_1 x(1) + a_0 x(0) = b(k)$$

式中，k 为正整数，a_0，a_1，…，a_k 均为常数。常系数线性差分方程的解法之一，可以通过 z 变换来取得。MATLAB 提供 z 变换函数 ztranst 和反变换函数 iztrans。z 变换的书写格式为

F = ztrans(f)

z 变换的定义为
$$F(z) = \sum_{0}^{\infty} \frac{f(n)}{z^n}$$

该式是将符号变量函数 f 进行 z 变换成 F，f 的默认变量为 n，而 F 的默认变量为 z。若 $x(n)$的 z 变换为 $x(z)$，则 $x(n+1)$的 z 变换为

$$Z[x(n+1)] = zx(z) - zx(0)$$

大写的 Z 表示对函数进行 z 变换运算，小写 z 则表示 z 运算了。

k 阶差分的 z 变换为 $Z[x(n+k)] = z^k x(z) - z^k x(0) - z^{k-1} x(1) - \cdots - zx(k-1)$

z 的反变换定义为

$$f(n) = \frac{1}{2\pi i} \oint_{|z|=R} F(z) z^{n-1*} \mathrm{d}z$$

式中 $n = 1, 2, \cdots$

有关常用函数的 z 变换和反变换可查找数学手册。

用 z 变换求解差分方程的步骤为，将差分方程进行 z 变换，使差分方程变为代数方程。解出差分函数的 z 变换表达式。将 z 变换表达式分解成部分分式，再将部分分式反变换成符号表达式。下面举例予以说明。

【例 8-41】 已知齐次差分方程式为 $x(n+2) + 3x(n+1) + 2x(n) = 0$，初始条件为 $x(0) = 0$，$x(1) = 1$，求解差分方程式。

解：

对上述差分方程式进行 z 变换得

$$z^2 x(z) - z^2 x(0) - zx(1) + 3zx(z) - 3zx(0) + 2x(z) = 0$$

将初值代入并化简得

$$x(z) = \frac{z}{z^2 + 3z + 2} = \frac{z}{(z+1)(z+2)} = \frac{z}{z+1} - \frac{z}{z+2}$$

在 MATLAB 命令窗口进行 z 反变换如下：

```
≫syms z                              %设符号变
≫x = z/(z+1) - z/(z+2)               %输入 z 表达式
x =
z/(z+1) - z/(z+2)
≫iztrans(x)                          %将表达式进行 z 反变换,得差分方程的解
ans =                                %式中 n = 0,1,2,3…
```

(－1)^n－(－2)^n

【例 8-42】 已知差分方程式为 $x(n+2)=x(n+1)+x(n)$，初始条件为 $x(0)=1$，$x(1)=1$，求差分方程的解。

解:

对差分方程进行 z 变换，并代入初始条件得

$$x(z)=\frac{z^2}{z^2-z-1}$$

```
≫syms z c1 c2                                   %设符号变量
≫x = z^2/(z^2 - z - 1)                          %输入 z 变换表达式
≫solve(z^2 - z - 1)                             %解分母多项式的根
ans =
[  1/2*5^(1/2) + 1/2]
[- 1/2*5^(1/2) + 1/2]
≫a1 = ans(1); a2 = ans(2);                      %赋值给 a1、a2
≫x = c1*z/(z + a1) + c2*z/(z + a2);             %分解部分分式
≫c1 = (1 - 1/sqrt(5))/2; c2 = (1 + 1/sqrt(5))/2 %解得 c1、c2
≫n = 0:10;                                      %设置 n 向量
≫a1 = (1 + sqrt(5))/2                           %设置 a1、a2 为标量
≫a2 = (1 - sqrt(5))/2;
≫x1 = c1*a1.^n + c2*a2.^n                       %对 x 进行反变换,即为差分方程的
                                                  解
x1 =                                            %差分方程解在 n = 0:10 时的值
                                                  这就是著名的 Fibonacci 数列
  Columns 1 through 8
    1.0000    1.0000    2.0000    3.0000    5.0000    8.0000    13.0000
    21.0000
  Columns 9 through 11
    34.0000    55.0000    89.0000
```

8.10 函数计算器

funtool 是 MATLAB 一个十分有用的可视化的函数计算器。当点击相关按钮，它能巧妙地显示单变量函数。作为例子，funtool 能够绘制由你指定二个函数的和、差、积、商。Funtool 包含有函数寄存器，它允许你存储函数，以便以后使用。

在 MATLAB 命令窗口输入 funtool，即出现函数计算器的图形窗口。如图 8-7～图 8-9 所示。图 8-7 显示函数 f 的图形，在默认的情况下显示 f = x。图 8-8 显示函数 g 的图形，在缺省的情况下，显示 g = 1。图 8-9 则为控制面板的图形，在控制面板上，装满了控制按钮，使控制面板具有编辑的功能。

在控制面板的顶部，包含有 4 组用来编辑表达式的字段框。

f——用来编辑或显示函数 f 的一个符号函数表达式；

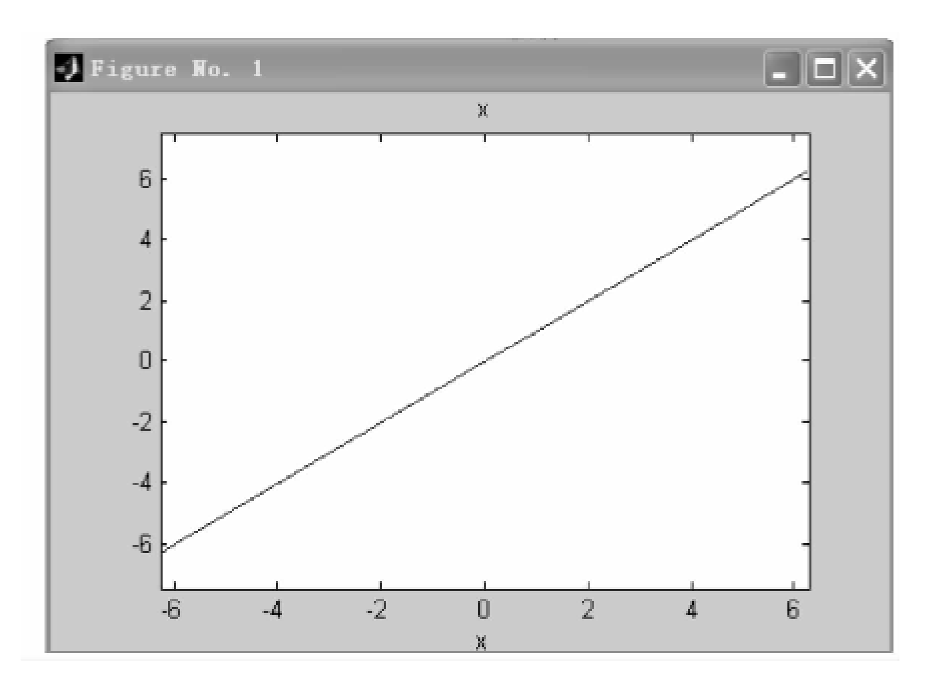


图 8-7 图形计算器的窗口 1，f 图形窗口

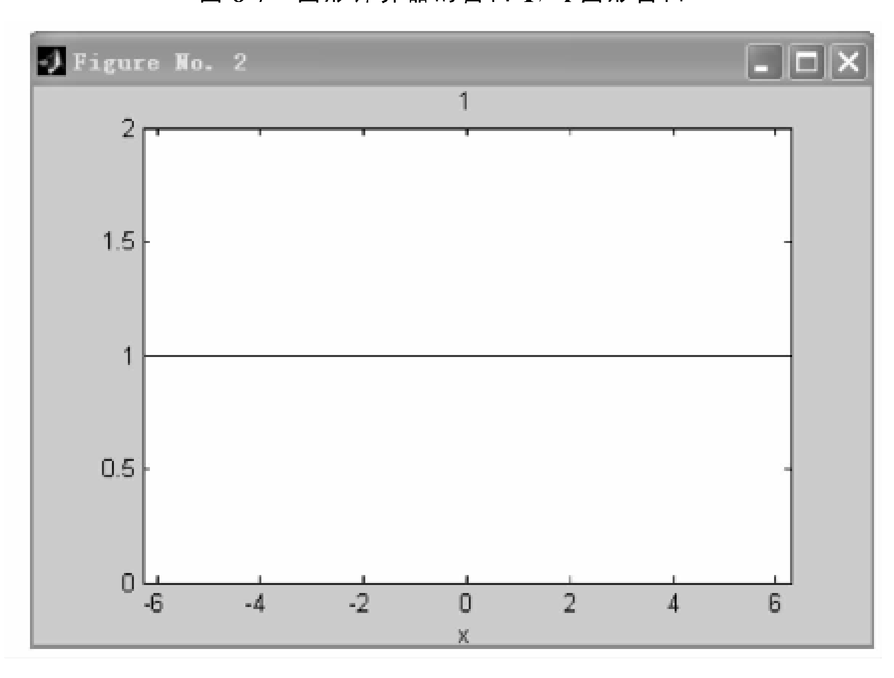


图 8-8 图形计算器的窗口 2，函数 g 的图形窗口

g——用来编辑或显示函数 g 的一个符号函数表达式；

x——用来输入函数 f，g 的定义域，在默认的情况下为［-2pi，2pi］；

a——用来输入一个常数项。

Funtool 上的图形是随着函数表达式 f、g 的内容而改变的。这改变是通过控制面板上输入字段名的内容决定的。

控制面板上，成排的，并附有内容的按钮是用来执行或转换相应的操作。

df/dx——计算函数 f 对 x 的微商；

int f——计算函数 f 的不定积分；

simple f——化简函数 f 的表达式；

图 8-9　图形计算器的窗口 3，控制面板

num f——提取函数 f 的分子表达式，并赋予 f;

den f——提取函数 f 的分母表达式，并赋予 f;

1/f——计算函数 f 的倒数函数，并赋予 f;

finv——计算函数 f 的逆函数，并赋予 f。

操作器在执行 int f 和 fin f 时可能会失败，假如相应的表达式是不存在。

控制面板的第二行按钮是用来计算函数 f 和常数 a 之间的运算，如纵坐标平移，横坐标平移，坐标轴的比例放大和缩小以及函数 f 的乘幂等。

f + a——计算 f – a;

f – a——计算 f – a;

f* a——计算 f* a;

f/a——计算 f/a;

f^a——计算 f^a;

f(x + a)——计算 f(x + a);

f(x* a)——计算 f(x* a)。

控制面板第三行中 1～4 个按钮用来替换函数 f 具有 f 和 g 组合的功能，余下的按钮是求复合函数以及 f 与 g 的互相交换。具体分配如下：

f + g——计算 2 函数 f，g 之和，并赋予 f;

f – g——计算 2 函数 f，g 之差，并赋予 f;

f* g——计算 2 函数 f，g 之积，并赋予 f;

f(g)——计算复合函数 f(g);

g = f——将函数 f 赋予函数 g;

swap——将函数 f 与 g 的表达式互相交换。

控制面板的第四行中按钮功能如下：

insert——将当前函数图形和表达式存入函数计算器中，作为典型函数图形;

cycle——依次演示函数计算器中，内存的典型函数图形；

delete——删除当前图形 1 窗口的典型函数图形；

reset——使函数计算器复位，复位到函数计算器初始状态；

help——提供函数计算器的使用帮助；

demo——自动演示函数计算器中，内存的典型函数图形；

Close——关闭函数计算器。

图 8-10、图 8-11 所示是 $f = \cos(x^3)/(1+x^2)$，$x = [-2\pi, 2\pi]$ 和 $g = 1/2\exp(-x)\sin(10x)$，$x = [0,4]$ 的图形演示。

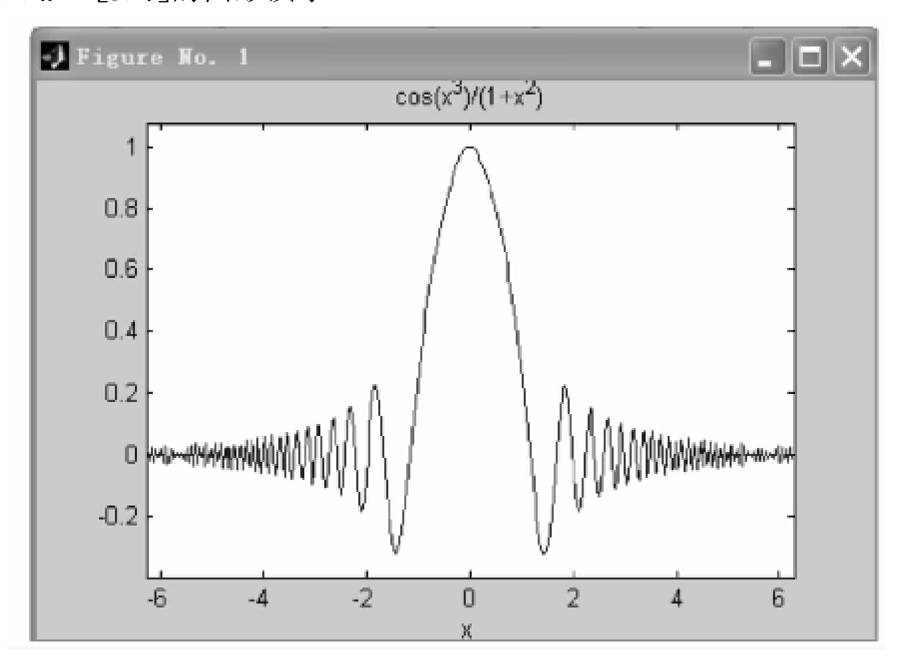


图 8-10　函数 f 的图示

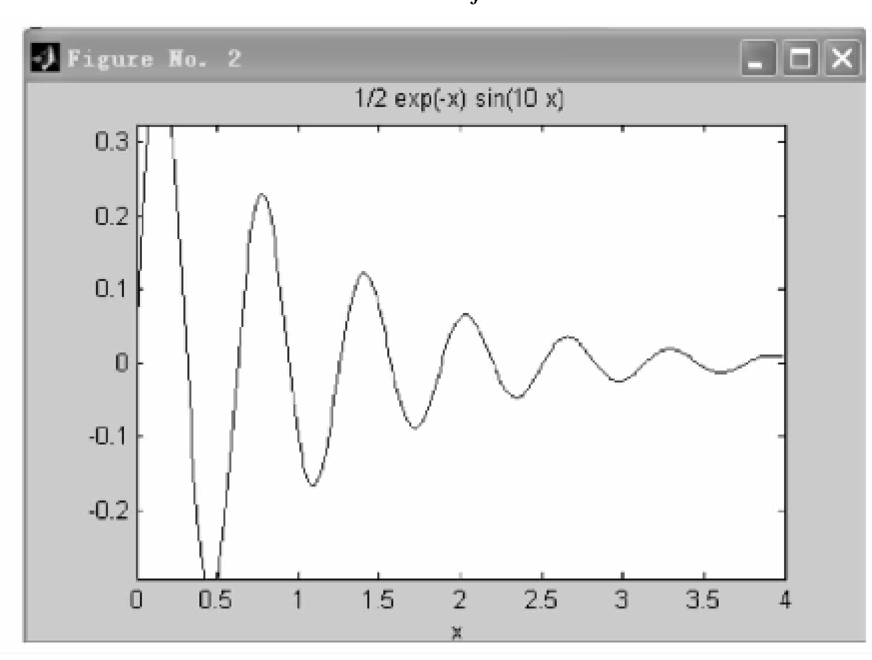


图 8-11　函数 $g = 1/2\exp(-x)\sin(10x)$ 的图示

8.11 泰勒级数计算器（Taylor tool）

Taylortool是教育上有用的工具，它设计的目的，是为了帮助读者如何用单变量的幂级数$T(x)$即Taylor级数，去近似函数$f(x)$。Taylortool计算器的图形如图8-12所示，它由图形显示框、Taylor级数展开表达式框、原函数f(x)表达式输入框、级数选择框、级数增加、减少按钮，常数项框，自变量上、下限框，以及Help，Reset，Close按钮组成。

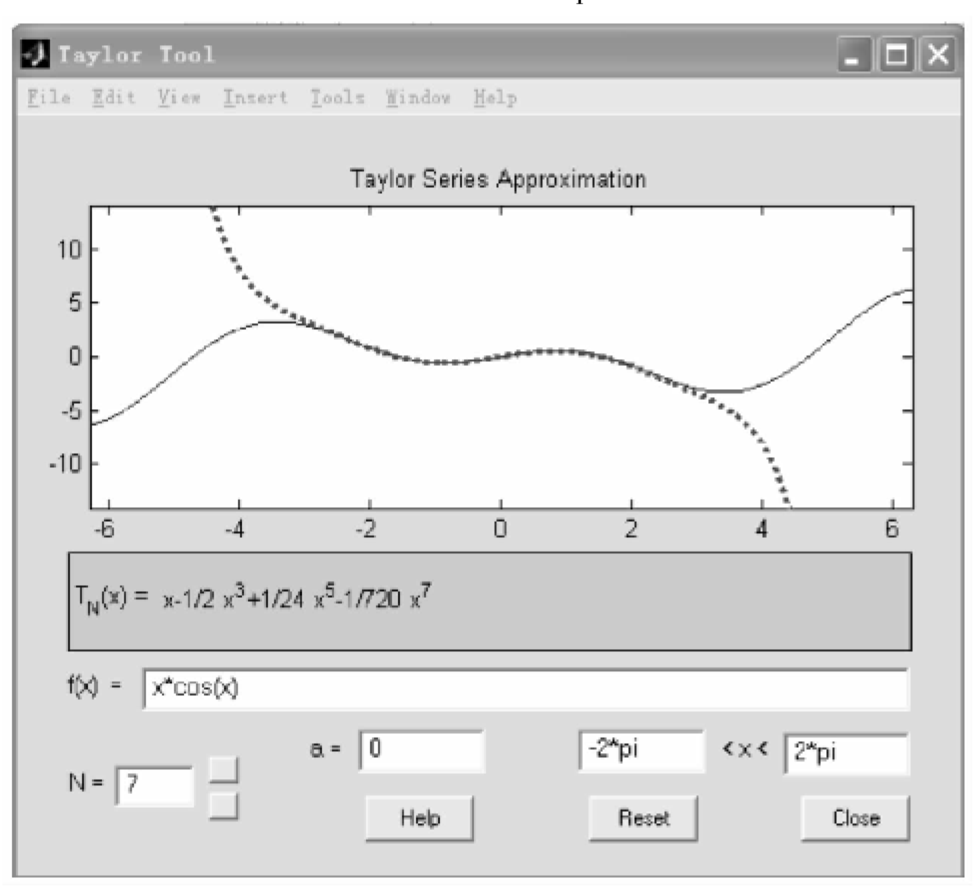


图 8-12 Taylor级数计算器

图形显示框，显示在自变量区间内，函数$f(x)$和$T(x)$的图形。蓝色实线表示函数$f(x)$，红色虚线表示$T(x)$。

Taylor级数展开表达式框，显示Taylor级数的近似表达式。它展开的项数，由级数选择器旁，点击增大或减小按钮来实现。

f(x)表达式输入框，用来输入函数f(x)的表达式。

常数项a用来显示函数f(x)的中心位置。

自变量上、下限框用来输入自变量的下限和上限。上述数据你可以通过Enter键来输入，通过Tab键来转移输入框。

点击Help按钮，则提供帮助文本。Reset按钮使Taylortool计算器复位。Close则关闭Taylortool计算器。

【例 8-43】 已知函数$f(x)=\dfrac{1}{1+x^2}$，求其Taylor级数展开式，展开到16次。

解：

如图 8-13 所示：

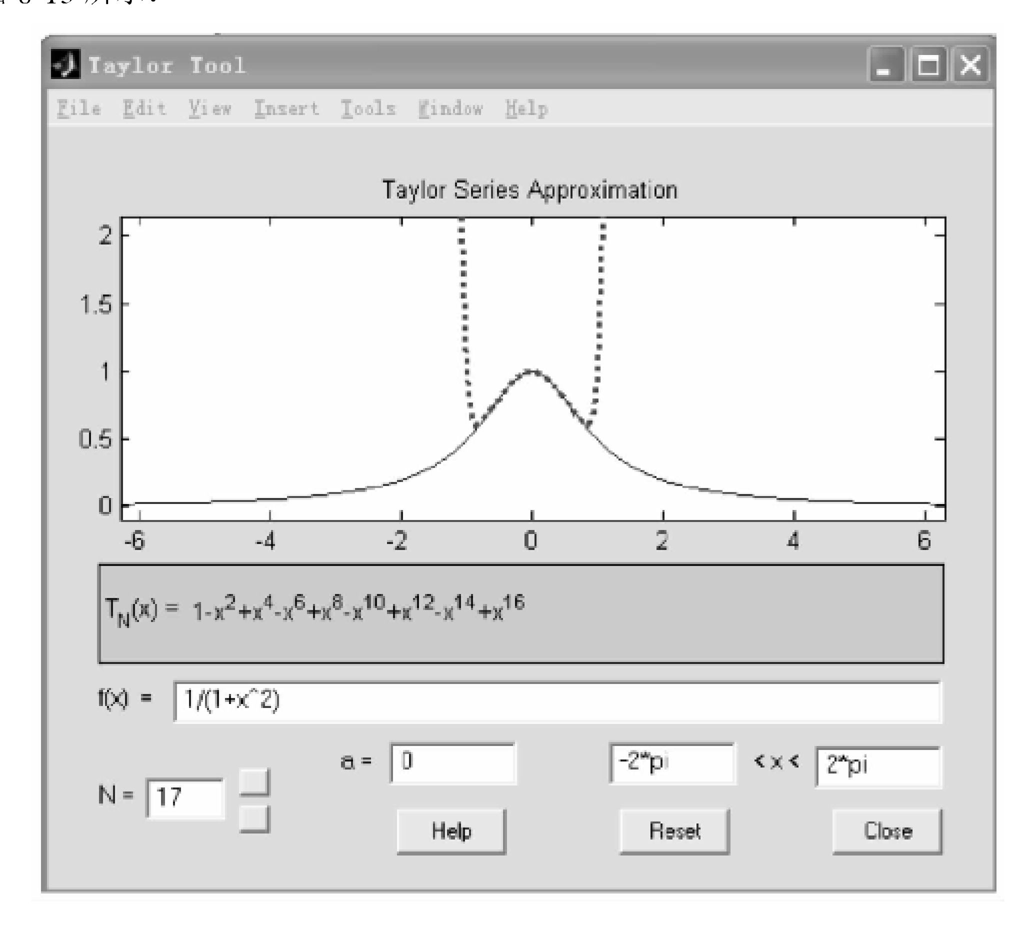


图 8-13　$f(x) = 1/(1 + x^2)$展开成 Taylor 级数

(1) 打开 Taylortool;

(2) 输入f(x) = 1/(1 + x^2)；

(3) 选择 N = 17;

(4) T(x) = 1 − x^2 + x^4 − x^4 + x^6 − x^8 + x^10 − x^12 + x^14 − x^16;

　　当 x < 1 时收敛。

第 8 章习题

8-1　用微分函数 diff，求下列函数的微分

(1) $y = x\sqrt{1 + x^2}$　　　(2) $y = \tan x$

(3) $y = x\lg x$　　　(4) $y = \sin x \sin(2x) \sin(3x)$

8-2　求高阶导数

$y = \exp x / x$，求 y^{10}

8-3　用积分函数 int，求下列函数的积分

(1) $\int x^2 a\cos(x)\mathrm{d}x$　　　(2) $\int \lg(x + \sqrt{1 + x^2})\mathrm{d}x$

8-4　求下列函数的定积分

(1) $\int_0^{\pi} (x\sin x)^2 \mathrm{d}x$　　　(2) $\int_{-1}^{1} \frac{x\,\mathrm{d}x}{x^2 + x + 1}$

8-5　用辛普森积分法函数 quad 计算积分 $\int_0^{\pi/2} \sin x \sin(2x)\sin(3x)\,dx$

8-6　求下列函数的极限

（1）$\lim\limits_{x \to 1} \dfrac{x^3 - 3x + 2}{x^4 - 4x + 3}$　　（2）$\lim\limits_{x \to 0} \dfrac{\sin 5x}{x}$

8-7　用常微分方程符号解函数 dsolve 解下列微分方程式。

（1）$\dfrac{dy}{dx} + 3y = \dfrac{x}{y}$　　初值条件为 $y(0) = 0$

（2）$\dfrac{d^2 y}{dx^2} + 2\dfrac{dy}{dx} + y = x$　　初值条件为 $y(0) = 0$，$y'(0) = 1$

8-8　将下列三角函数展开成 Taylor 级数，取其 9 阶以内。

$Y = \sin x \sin(2x)\sin(3x)$

第9章　MATLAB在工程最优化中的应用

最优化是每个人，每个单位所希望实现的事情。对于产品设计者来说，是考虑如何用最少的材料，最大的性能价格比，设计出满足市场需要的产品。对于企业的管理者来说，则是如何合理使用劳力，充分使用现有设备，减少库存，降低能耗，降低成本，以实现企业的最大利润。

对于投资者来说，要考虑在市场不确定的情况下，掌握投资方向，规避风险，实现投资最优化。对于股民，总希望以最低的价格买入股票，而以最高价格卖出股票，以获取最大收益。对于消费个人，则考虑在最小支出的情况下，如何取得最大需求。对于旅程问题，人们总希望从甲地到乙地的路程最短。如果为了赶时间则希望旅程中所花时间最少。如果时间不是主要问题，则希望旅程中所花的费用最小。这就引出了距离最优，时间最优和消耗最优的问题。

最优化历来是数学家所关心的问题，牛顿（Newton）、莱布尼茨（Leibniz）在17世纪创造了微积分，通过函数的微商等于零，求函数的极值问题。拉格朗日（Lagrange）采用拉格朗日乘子法解决了多元函数求极值问题。但是数学上能解决的最优化，都是在条件确定的情况下取得的。然而，现实世界的丰富多彩就是因为它的不确定性。通常在你做出决策前，你无法掌握有关最优化对象当前的全部信息和近期的信息，更难以预见将来的信息。因此你的多步最优化的任务就难以实现，往往只能走一步，看一步，实现次于最优，仍至无法实现最优化。只有少数精英，他能运用他的经验、智慧和锐利的直觉预见当前尚未显现的现象，做出多步最优化策略。

MATLAB优化工具箱有丰富的内容，其中常用优化函数见表9-1。

表9-1　MATLAB常用优化函数

函　数　名	说　　明
fminbnd	有边界的标量最小化
fminunc	无约束非线性最小化
fminsearch	无约束非线性最小化搜索
linprog	线性规划，具有等式约束或不等式约束
quadprog	二次规划，具有等式约束或不等式约束
fmincon	条件约束非线性最小化
fgoalattain	多目标达到课题
fminmax	最大最小化
fseminf	半无限最小化
lsqlin	有约束线性最小二乘解

本章要介绍的最优化，其中包括无约束最优化，有约束最优化，有约束最优化的图解法，多级铁心设计的最优化，漏斗设计的最优化，旅程时间最小问题和二次最优化等。对于

线性规划，由于内容较多，则放在第 10 章介绍。

9.1 无约束的最优化

提到最优化，首先要有个目标函数，它应该是可以计量的，或者可以用数学表达式表示的。寻求它的最大值或最小值就成为最优化的任务。假设目标函数为$f(x)$，其中 x 为向量，则在无约束条件下的最优化，只要使 $f(x)$关于向量 $\boldsymbol{x}$ 的各个元素的偏微商为零，解出向量 $\boldsymbol{x}^*$，并核对$f(\boldsymbol{x}^*)$是否是最优值即可。为了将求极大的问题统一为求极小问题，只要把求极大值的目标函数改为负值，去解极小值问题，解出后，再把目标值符号改回来即可。

对于单变量函数寻求极值，则比较间单，可以对$f(x)$取一阶微商等于零，求解方程式即可，若在 $\boldsymbol{x}^*$ 处二阶微商大于零则有极大值，小于零则有极小值。

MATLAB 是用函数 fminunc，来寻找无约束条件下多变量函数的最小值。fminunc 的书写格式为

```
    x = fminunc(fun, x0)
    x = fminunc(fun, x0, options)
[x, fval] = fminunc(…)
[x, , fval, exitflag] = fminunc(…)
[x, fval, exitflag, output] = fminunc(…)
```

式中，x、x0 为向量，x 为解向量，x0 为初始值向量，fun 为函数表达式，options 为选项，用来选择允许计算误差、迭代次数、算法等，详见 optimset 函数*。exitflag 为退出标注。exitflag > 0 函数收敛于解 x；exitflag = 0，迭代次数超过；exitflag < 0，则函数值不收敛。output 则输出解题信息，有关迭代次数，函数值计算次数，在 x 处的梯度范数和所用算法等。

与求解无约束条件下多变量函数的最小值函数相类似的函数为 fminsearch，它用于多变量最小值的搜索，它的书写格式与上述相似

```
    x = fminsearch(fun, x0)
    x = fminsearch(fun, x0, options)
[x, fval] = fminsearch(…)
[x, , fval, exitflag] = fminsearch(…)
[x, fval, exitflag, output] = fminsearch(…)
```

读者可以参照使用。

* optimset 函数

optimset 函数是用来设置最优化选项（Options）用。Options 的调用格式为

options = optimset（'param1'，value1，'param2'，value2，…）

式中，param1，param2…优化参数名，value1，value2…为优化参数对应的值，由 optimset 列表规定。它的内容显示如下，列表左侧为优化参数名，右侧方括号内为优化参数名的选项。

```
≫optimset
    DerivativeCheck: [on | {off}]
```

Diagnostics: [on | {off}]
DiffMaxChange: [positive scalar {1e－1}]
DiffMinChange: [positive scalar {1e－8}]
Display: [off | iter | notify | final]
GoalsExactAchieve: [positive scalar | {0}]
GradConstr: [on | {off}]
GradObj: [on | {off}]
Hessian: [on | {off}]
HessMult: [function | {[]}]
HessPattern: [sparse matrix | {sparse(ones(NumberOfVariables))}]
HessUpdate: [dfp | gillmurray | steepdesc | {bfgs}]
Jacobian: [on | {off}]
JacobMult: [function | ([])]
JacobPattern: [sparse matrix | {sparse(ones(Jrows,Jcols))}]
LargeScale: [{on} | off]
LevenbergMarquardt: [on | off]
LineSearchType: [cubicpoly | {quadcubic}]
MaxFunEvals: [positive scalar]
MaxIter: [positive scalar]
MaxPCGIter: [positive scalar | {max(1,floor(numberOfVariables/2))}]
MeritFunction: [singleobj | multiobj]
MinAbsMax: [positive scalar | {0}]
NonlEqnAlgorithm: [{dogleg} | lm | gn]
PrecondBandWidth: [positive scalar | {0} | Inf]
Simplex: [on | {off}]
TolCon: [positive scalar]
TolFun: [positive scalar]
TolPCG: [positive scalar | {0.1}]
TolX: [positive scalar]
TypicalX: [vector | {ones(NumberOfVariables,1)}]

上述的优化参数说明见表 9-2，在表的适用范围一栏中，代号 M 为中规模算法，L 为大规模算法，B 表示二者都适用。

表 9-2 优化参数说明与适用范围

项号	参数名	说明	适用范围
1	DerivativeCheck	用户提供微商与有限差分微商的比较	M
2	Diagnostics	输出诊断信息有关函数的最优化	B
3	DiffMaxChange	有限差分微商的最大变化量	M
4	DiffMinChange	有限差分微商的最小变化量	M

（续）

项　号	参 数 名	说　明	适用范围
5	GoalsExactAchieve	精确达到目标数	M
6	GradConstr	由用户定义的非线性约束的梯度	M
7	GradObj	由用户定义的目标函数的梯度	B
8	Hessian	由用户定义的目标函数的海森矩阵	L
9	HessMult	由用户定义的多变量函数的海森矩阵	L
10	HessPattern	有限差分的海森矩阵的稀疏类型	L
11	HessUpdate	全撒 - 牛顿更新方案	M
12	Jacobian	由用户定义的目标函数的雅可比矩阵	B
13	JacobMult	由用户定义的多变量函数的雅可比矩阵	B
14	JacobPattern	有限差分的雅可比矩阵的稀疏类型	B
15	LargeScale	假如可能使用大规模算法	B
16	LevenbergMarquardt	选择 Levenberg - Marquardt 法覆盖高斯 - 牛顿法	M
17	LineSearchType	选择线性搜索算法	M
18	MaxPCGIter	最大允许 PCG 迭代次数	L
19	MeritFunction	使用多目标/最小最大的价值函数	M
20	MinAbsMax	F（x）在最差情况下的绝对值到最小的个数	M
21	PrecondBandWidth	PCG 的上限宽度	L
22	TolCon	约束变量最终允许误差	B
23	TolPCG	PCG 迭代最终允许误差	L
24	TypicalX	典型 x 的值	L
25	Display	显示水平，“off”为显示关；“iter”显示迭代次数；“final”显示最终输出；“notify”则在函数不收敛时显示输出	B
26	MaxFunEvals	允许函数估值的最大次数	M
27	MaxIter	最大允许迭代次数	B
28	TolFun	函数值最终允许误差	B
29	TolX	变量 x 最终允许误差	B

由于最优化参数设置的参数较多。为了省事，可以使用默认设置，默认设置时可以在 optimset 函数内填写被调用优化函数名即可。例如 options = optimset（'fminbnd'），这语句返回一个最优化的选项结构，它包含所有的参数名和默认值，相关于优化函数 fminbnd。最优化设置也可以在原有的基础上进行修改或补充。例如从原有的基础上添加允许函数的误差为 1e - 8，则可以做如下设置：options = optimset（options,'TolFun',1e - 8）即可。其中 options 为原先的设置，而 TolFun 为添加设置。

【例 9-1】　已知 $y = 1 + 2x - x^2$，求最大值。

解：

由于 MATLAB 的最优化函数中，只有求最小值，故把题设函数改成负值来计算。

```
>>fun = inline('x^2 - (1 + x)')                  %设置内联函数 fun,把函数改为负值
```

```
fun =
    Inline function:
    fun(x) = x^2 - (1 + x)
≫options = optimset('fminunc')                              %无约束最小化的默认设置
≫options = optimset(options,'largescale','off');            %添加不执行大规模算法
≫[x,fval,exitflag,output] = fminunc(fun,0.5,options)        %计算最优值
Optimization terminated successfully:                        %最优化成功
Search direction less than 2* options.TolX
x =                                                          %x 的解
    0.5000
fval                                                         %最优函数值,这时应把符号反过来,正解应为 1.25
  -1.2500
exitflag =                                                   %退出原因,函数收敛
    1
output =                                                     %输出解算过程信息
      iterations: 1                                          %迭代次数
      funcCount: 2                                           %函数计算次数
      stepsize: 1                                            %步长大小
    firstorderopt: 2.2204e-008                               %第一次优化后误差
    algorithm: 'medium-scale: Quasi-Newton line search'      %算法:中规模
```

【例 9-2】 在变压器铁心截面计算中，经常用到多级矩形叠片，使铁心在圆内占空比最大。求三级铁心中，每级铁心的宽度。假定圆的半径为 1。

解：

由于对称关系，只要分析 1/4 圆就可以了。由于铁心的对称关系，第二级铁心必定在 45°上，如图 9-1 所示。

1/4 铁心面积

$$A_3 = \sin(zeta)\cos(zeta) + \sin(\pi/4 - zeta)\cos(\pi/4) + (\cos(zeta) - \sin(\pi/4))\sin(zeta)$$
$$= 2\sin(zeta)\cos(zeta) - \sqrt{2}\sin(zeta) + 1/2$$

式中，zeta 为第一级铁心的矩形顶点到原点的连线与 x 轴的夹角。

在 MATLAB 命令窗口输入如下程序：

```
≫x0 = pi/8;                                                  %设初值
≫fun = inline('sqrt(2)* sin(zeta) - 2* sin(zeta)* cos(zeta) - 1/2')
fun =                                                        %设内联函数取 A3 的负值
    Inline function:
    fun(zeta) = sqrt(2)* sin(zeta) - 2* sin(zeta)* cos(zeta) - 1/2
≫options = optimset('fminunc')                              %无约束最小化的默认设置
≫options = optimset('largescale','off');                    %不执行大规模算法
```

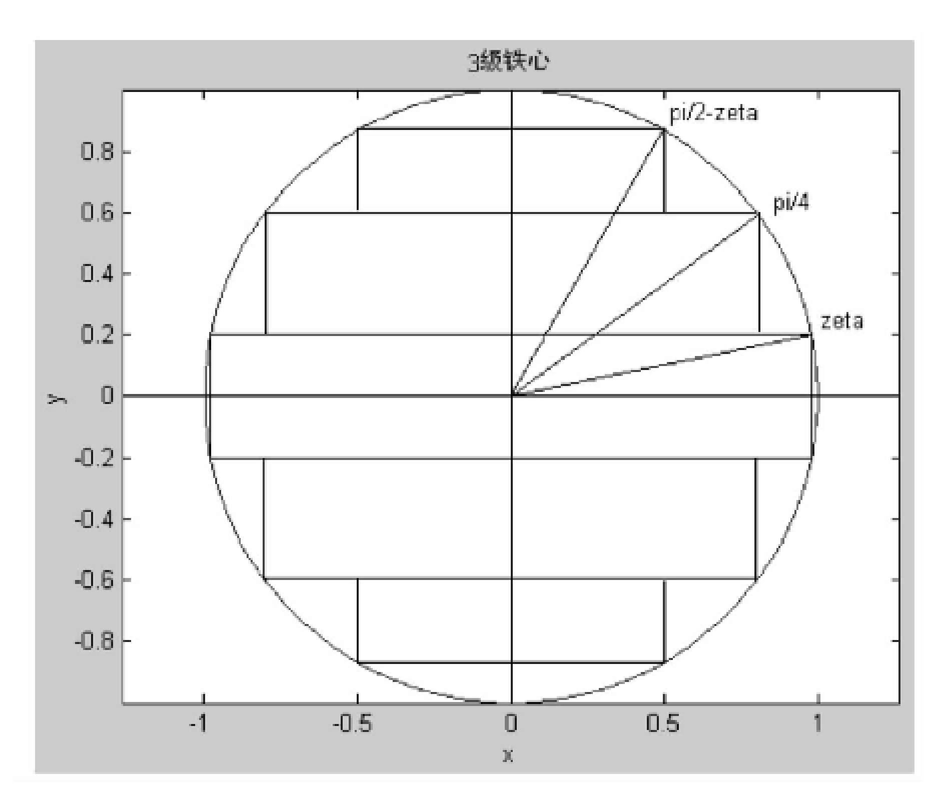


图 9-1　三级铁心计算图

```
>>[zeta,fval] = fminunc(fun,x0,options)          %计算 fun 的最小值,即 A3 的最大值
Optimization terminated successfully:             %最优化成功
Current search direction is a descent direction, and magnitude of
directional derivative in search direction less than 2* options.TolFun
zeta =                                             %第一级铁心的角度
    0.4379
fval =                                             %面积最大值,取 fval 的负值
    -0.6684
>>a1 = cos(zeta)                                   % 第一级铁心宽度应为 2rcos(zeta),现考虑
                                                     2r = 1,即直径为 1,则 cos(zeta)就成为铁心
                                                     长。下同
a1 =
    0.9056
>>a2 = cos(pi/4)                                   %第二级铁心宽度
a2 =
    0.7071
>>a3 = cos(pi/2 - zeta)                            %第三级铁心宽度
a3 =
    0.4240
>>kn = -fval/pi* 4                                 %3 级铁心的占空比
kn =
    0.8510
```

【例 9-3】 承接上例，当铁心级数为 4、5、6 级的情况。与现行的铁心计算的手册数据相比较。

解：

%4 级铁心的面积公式的函数 M 文件为

function A4 = core4(x)

A4 = 2* sin(x(1))* cos(x(2)) + sin(x(2)).^2 - 2* sin(x(1))* cos(x(1)) - 2* sin(x(2))* cos(x(2))

%式中，x(1)、x(2)为 1、2 级铁心的矩形顶点到原点的连线与 x 轴的夹角；3、4 级铁心的矩形顶点到原点的连线与 x 轴的夹角为pi/2 - x(2)，pi/2 - x(1)

%5 级铁心的面积公式的函数 M 文件为

function A5 = core5(x)

A5 = 2* sin(x(1))* cos(x(2)) + sqrt(2)* sin(x(2)) - 2* sin(x(1))* cos(x(1)) - 2* sin(x(2))* cos(x(2)) - 1/2

%式中，x(1)、x(2)为 1、2 级铁心的矩形顶点到原点的连线与 x 轴的夹角；x(3) = pi/4；4、5 级铁心的矩形顶点到原点的连线与 x 轴的夹角为pi/2 - x(2)，pi/2 - x(1)

%6 级铁心的面积公式的函数 M 文件为

function A6 = core6(x)

A6 = 2* sin(x(1))* cos(x(2)) + 2* sin(x(2))* cos(x(3)) + sin(x(3))^2 - 2* sin(x(1))* cos(x(1)) - 2* sin(x(2))* cos(x(2)) - 2* sin(x(3))* cos(x(3))

%式中，x(1)、x(2)、x(3)为 1、2、3 级铁心的矩形顶点到原点的连线与 x 轴的夹角；4、5、6 级铁心的矩形顶点到原点的连线与 x 轴的夹角为pi/2 - x(3)、pi/2 - x(2)、pi/2 - x(1)。

4 级铁心计算结果如下：

```
>>options = optimset('fminunc')                        %无约束最小化的默认设置
>>options = optimset(options,'largescale','off');      %添加不执行大规模算法
>>[x,fval,exitflag,output] = fminunc(@core4,[pi/8;pi/6],options)
                                                        %计算函数 core4 的最小值，初始值为
                                                          [pi/8;pi/6] 向量
Optimization terminated successfully:
Current search direction is a descent direction, and magnitude of
directional derivative in search direction less than 2* options.TolFun
x =                                                     %最优解
    0.3673
    0.6516
fval =                                                  %极小值，对应铁心面积为最大值
     -0.6959
exitflag =                                              %迭代运算收敛
    1
output =
       iterations: 3                                    %迭代次数
```

```
        funcCount: 16                                          %函数估值次数
     firstorderopt: 1.5213e-005                                %第一次优化后误差
     algorithm: 'medium-scale: Quasi-Newton line search'       %中规模线性搜索算法
ans =                                                          %第一级铁心宽度
     0.9333
>>cos(x(2))
ans =                                                          %第二级铁心宽度
     0.7951
>>sin(x(2))
ans =                                                          %第三级铁心宽度
     0.6065
>>sin(x(1))
ans =                                                          %第四级铁心宽度
     0.3591
>>k5 = -fval/pi*4                                              %4级铁心占空比
k5 =
     0.8860
```

5级铁心计算结果如下：

```
>>[x,fval,exitflag,output] = fminunc(@core5,[pi/8;pi/6],options)
                                                               %计算函数 core5 的最小值，初
                                                                 始值为向量[pi/8;pi/6]
Optimization terminated successfully:
Current search direction is a descent direction, and magnitude of
directional derivative in search direction less than 2*options.TolFun
x =                                                            %最优解
     0.3192
     0.5628
fval =                                                         %极小值，对应铁心面积为最大
                                                                 值
     -0.7131
exitflag =
     1
output =
        iterations: 3
         funcCount: 16
          stepsize: 1.0003
     firstorderopt: 0.0017
         algorithm: 'medium-scale: Quasi-Newton line search'
>>cos(x(1))
```

```
ans  =                                              %第一级铁心宽度
     0.9495
≫cos(x(2))
ans =                                               %第二级铁心宽度
     0.8457
≫cos(pi/4)
ans =                                               %第三级铁心宽度
     0.7071
≫sin(x(2))
ans =                                               %第四级铁心宽度
     0.5336
≫sin(x(1))
ans  =                                              %第五级铁心宽度
     0.3139
≫k5 = -fval/pi*4                                    %5 级铁心占空比
k5  =
     0.9079
```

6 级铁心计算结果如下：

```
≫[x,fval,exitflag,output] = fminunc(@core6,[pi/8;pi/6;pi/4],options)
                                                    %计算函数 core5 的最小值,初始值
                                                      为向量[pi/8;pi/6;pi/4]
Optimization terminated successfully:
Current search direction is a descent direction, and magnitude of
directional derivative in search direction less than 2*options.TolFun
x =                                                 %最优解
     0.2840
     0.4988
     0.6919
fval =                                              %极小值,对应铁心面积为最大值
     -0.7248
exitflag =
     1
output =
        iterations: 4
         funcCount: 26
          stepsize: 0.6496
     firstorderopt: 0.0012
         algorithm: 'medium-scale: Quasi-Newton line search'
≫cos(x(1))                                          %第一级铁心宽度
```

```
ans =
      0.9599
>>cos(x(2))                                    %第二级铁心宽度
ans =
      0.8782
>>cos(x(3))                                    %第三级铁心宽度
ans =
      0.7700
>>sin(x(3))                                    %第四级铁心宽度
ans =
      0.6380
>>sin(x(2))                                    %第五级铁心宽度
ans =
      0.4784
>>sin(x(1))                                    %第六级铁心宽度
ans =
      0.2802
>>k6 = -fval/pi*4                              %6级铁心占空比
k6 =
      0.9228
```

现行多级铁心设计参数见表 9-3，用 MATLAB 设计多级铁心见表 9-4。

表 9-3 现行多级铁心设计表

级 数	k_n	B_1	B_2	B_3	B_4	B_5	B_6
2	0.789	0.85	0.505				
3	0.851	0.905	0.707	0.424			
4	0.866	0.935	0.8	0.6	0.355		
5	0.910	0.95	0.847	0.707	0.572	0.312	
6	0.930	0.955	0.87	0.77	0.64	0.495	0.3

注：表中 $B_1 \sim B_6$ 为铁心宽度与直径之比，kn 为铁心截面与圆面积之比。

表 9-4 用 MATLAB 设计多级铁心表

级 数	k_n	B_1	B_2	B_3	B_4	B_5	B_6
2	0.7869	0.8506	0.5257				
3	0.8510	0.9056	0.7071	0.424			
4	0.8860	0.9333	0.7951	0.6065	0.3591		

（续）

级　数	k_n	B_1	B_2	B_3	B_4	B_5	B_6
5	0.9079	0.9495	0.8457	0.7071	0.5336	0.3139	
6	0.9228	0.9599	0.8782	0.7700	0.6380	0.4784	0.2802

注：同表 9-3。

比较上述二表，对于 2、3 级铁心，二表的数据是一致的。对于 4～6 级则有细小差别。

9.2　具有约束条件的最优化

最优化往往带有约束条件，例如从约束的性质上分，有时间约束、成本约束、劳动力约束、设备约束、材料约束、重量约束等。从数学上分有线性约束和非线性约束；有等式约束和不等式约束。最经典的具有约束条件的最优化问题是圆柱体表面积为常数的情况下，求圆柱体体积最大的问题。

圆柱体的表面积为

$$S = 2\pi r(r + h)$$

式中，r 为圆柱体半径，h 为圆柱体之高。在本例中是作为等式约束条件。

而圆柱体的体积为

$$V = \pi r^2 h$$

用拉格朗日乘子法，很容易证明 $h = 2r$ 时圆柱体的体积最大。

下面先介绍一个简单的漏斗设计的例子。漏斗广泛用于加料系统中，将颗粒从大的端口倒入小的端口进行加工。漏斗的设计要求在漏斗用料一定的情况下，使漏斗的容积最大。

【例 9-4】　已知漏斗进料口的半径为 r，锥体的斜棱长为 R，锥体平面角的 1/2 为 x，漏斗是由圆形薄板，经剪去扇形后弯成。如图 9-2 所示，求在漏斗表面积 S 一定时，漏斗的容积最大。

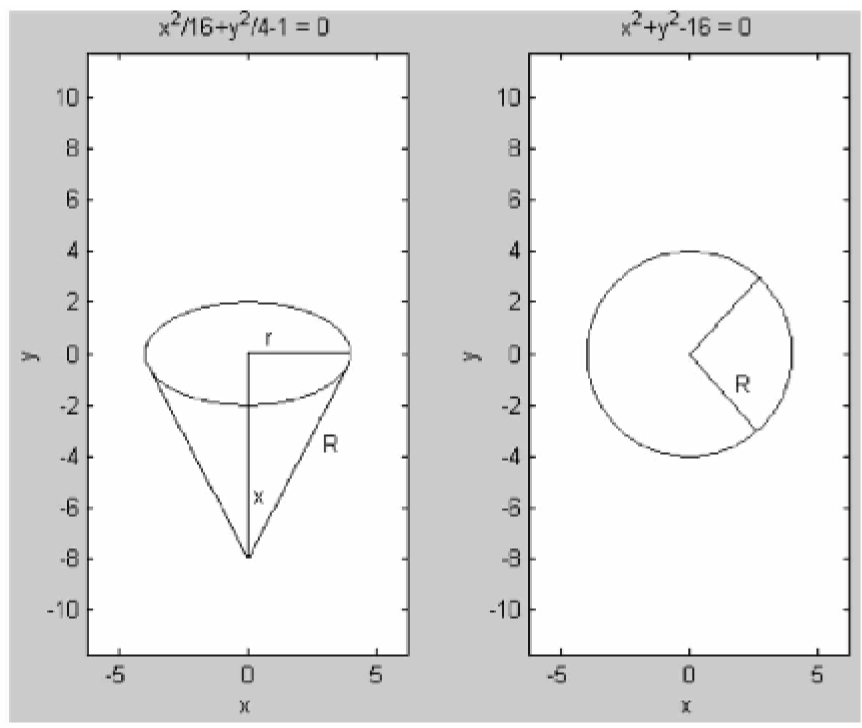


图 9-2　漏斗的尺寸及其构成

解：

%由于漏斗的容积是由两个变量 R、x 决定的，而约束条件为 S，故采用拉格朗日乘子法来求解。

%选择拉格朗日乘子为 lambda

```
>>clear                                         %清内存
>>syms x R p S lambda                           %设置符号变量,其中 p 为 π,若用 pi 代入,运算时会变成一长串数字,对显示不便
>>V = 1/3* p* R^3* sin(x)^2* cos(x) + lambda* (S - p* R^2* sin(x))
                                                %输入容积
V =
1/3* p* R^3* sin(x)^2* cos(x) + lambda* (S - p* R^2* sin(x))
>>eq1 = diff(V,'R')                             %对变量 R 求微商
eq1 =
p* R^2* sin(x)^2* cos(x) - 2* lambda* p* R* sin(x)
>>lambda = solve(eq1,'lambda')                  %使微商为零,解 lambda
lambda =
1/2* R* sin(x)* cos(x)
>>eq2 = diff(V,'x')                             %对变量 x 求微商,并将 lambda 代入,化简
eq2 =
2/3* p* R^3* sin(x)* cos(x)^2 - 1/3* p* R^3* sin(x)^3 - lambda* p* R^2* cos(x)
>>eq2 = 1/6* p* R^3* sin(x)* cos(x)^2 - 1/3* p* R^3* sin(x)^3
                                                %用 lambda 代入,并化简结果
eq2 =
1/6* p* R^3* sin(x)* cos(x)^2 - 1/3* p* R^3* sin(x)^3
>> x = solve(eq2)                               %解 eq2 中的变量 x
x =                                             %零解和负解都不适合,所以正解为 x(2)

[                    0]
[  atan (1/2* 2^ (1/2))]
[- atan (1/2* 2^ (1/2))]
>>x1 = double (x (2))                           %把字符串数字转换成双精度数
x1 =
    0.6155
>>x11 = x1/pi* 180                              %折算成度数
x11 =
    35.2644
>>x12 = 2* x11                                  %漏斗的最优中心角
```

x12 =

70.5288

MATLAB 中对具有约束条件的最优化函数为 fmincon，它的最优化的描述如下：

$\min\limits_{x} f(x)$，$f(x)$为目标函数，它受约束于下面的式子：

c(x) < = 0 ——非线性函数不等式约束，x 为向量。

ceq = 0——非线性函数等式约束。

A* x < = b——线性不等式约束，A 为矩阵，b 为向量。

Aeq* x = beq——线性等式约束，Aeq 为矩，beq 为向量。

Lb < = x < = ub——最优值的区间，lb、ub 为向量。

Fmincon 的书写格式为

x = fmincon(fun,x0,A,b)——fun 为目标函数,用于线性约束,默认设置,x0 为初值

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub)——用于兼有等式和不等式约束

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon)——带有非线性函数约束

x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nonlcon,options)——带有选项,详见 optimset 对于最优化输出有以下几种格式：

```
[x,fval] = fmincon(…)                                          %增加最优值输出
[x,fval,exitflag] = fmincon(…)                                 %增加退出标记
[x,fval,exitflag,output] = fmincon(…)                          %增加输出报告
[x,fval,exitflag,output,lambda] = fmincon(…)                   %增加输出拉格朗日系数
[x,fval,exitflag,output,lambda,grad] = fmincon(…)              %增加梯度输出
[x,fval,exitflag,output,lambda,grad,hassian] = fmincon(…)      %增加赫森矩阵输出
≫[x,fval,exitflag,output] = fmincon(@coneq,x0,[],[],Aeq,beq,[0;0],[10;10],[],options)
```

【例 9-5】 今有一目标函数$f(x) = x_1^2 + x^2 - x_1 x_2 - 10x_1 - 4x_2 + 60$，线性约束条件为$x_1 + x_2 = 8$，求目标函数的最小值，$x_1$、$x_2$ 的上、下限为 [-5，10]，初值设$\boldsymbol{x}_0 = [0\ 0]$。

解：

```
%对目标函数编制函数文件如下：文件名为 coneq.m
function y = coneq(x)
y = x(1)^2 + x(2)^2 - x(1)* x(2) - 10* x(1) - 4* x(2) + 60;
```

在函数文件存盘后，在命令窗口输入如下：

```
≫Aeq = [1,1];beq = 8;A = [];b = [];                    %输入等式线性约束
≫x0 = [0;0];                                           %设置初值
≫lb = [-5;-5];ub = [10;10]                             %设置上、下限
≫options = optimset('largescale','off');               %设置选项，不执行大范围算法，即执行
                                                       %中范围算法
≫ [x, fval, exitflag, output] = fmincon (@coneq, x0, [], [], Aeq, beq, lb, ub, [],
options)
                                                       %计算最优
```

```
                                                    %解，缺项必需用空矩阵填
                                                     充，否则出错
Optimization terminated successfully:               %最优化成功
Search direction less than 2* options.TolX and
  maximum constraint violation is less than options.TolCon
Active Constraints:                                 %约束有效
     1
x =                                                 %解向量
     5.0000
     3.0000
fval =                                              %目标函数最小值
     17.0000
exitflag =                                          %退出原因，收敛
     1
output =                                            %输出解算信息
         iterations: 4                              %迭代次数，4次
          funcCount: 18                             %函数估值次数
           stepsize: 0.5000                         %步距
     algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
                                                    %算法中规模，线性搜索
     firstorderopt: []                              %一阶最优
      cgiterations: []                              %共轭梯度迭代次数
```

【例9-6】 考虑3段旅程所化时间最短问题。第一段为沙漠，长度（h）为10km，行走速度为每小时5km；第二段为河流，长度（h）为10km，渡船的速度每小时为10km；第三段为平地，长度（h）亦为10km，乘坐汽车，时速为40km，如图9-3所示。问从A点到B点，应采取何种走向，使3段行程所化的时间的总和最小。已知A、B的水平距离为$b=40$km。A、B的垂直距离为30km。

解：

设第一段从沙漠中的A点走到C点，在x轴方向行走了$x(1)$公里。第二段为河流，从C点走到D点，在x轴方向移动了$x(2)$公里，第三段为平地，从D点到终点B，在x轴方向移动了$x(3)$公里。目标函数为取时间最小

$$\min t=\frac{\sqrt{h^2+x^2(1)}}{v_1}+\frac{\sqrt{h^2+x^2(2)}}{v_2}+\frac{\sqrt{h^2+x^2(3)}}{v_3}$$

$$=\frac{h}{v_1}\sqrt{1+\frac{x^2(1)}{h^2}}+\frac{h}{v_2}\sqrt{1+\frac{x^2(2)}{h^2}}+\frac{h}{v_3}\sqrt{1+\frac{x^2(3)}{h^2}}$$

令 $x(1)/h\rightarrow x(1)$，$x(2)/h\rightarrow x(2)$，$x(3)/h\rightarrow x(3)$。考虑 $h/v1=2$，$h/v2=1$，$h/v3=0.25$，则

$$\min t=2\sqrt{(1+x^2(1)}+\sqrt{(1+x^2(2)}+0.25\sqrt{(1+x^2(3)}$$

约束条件为

$$x(1)+x(2)+x(3)=b/h=4$$

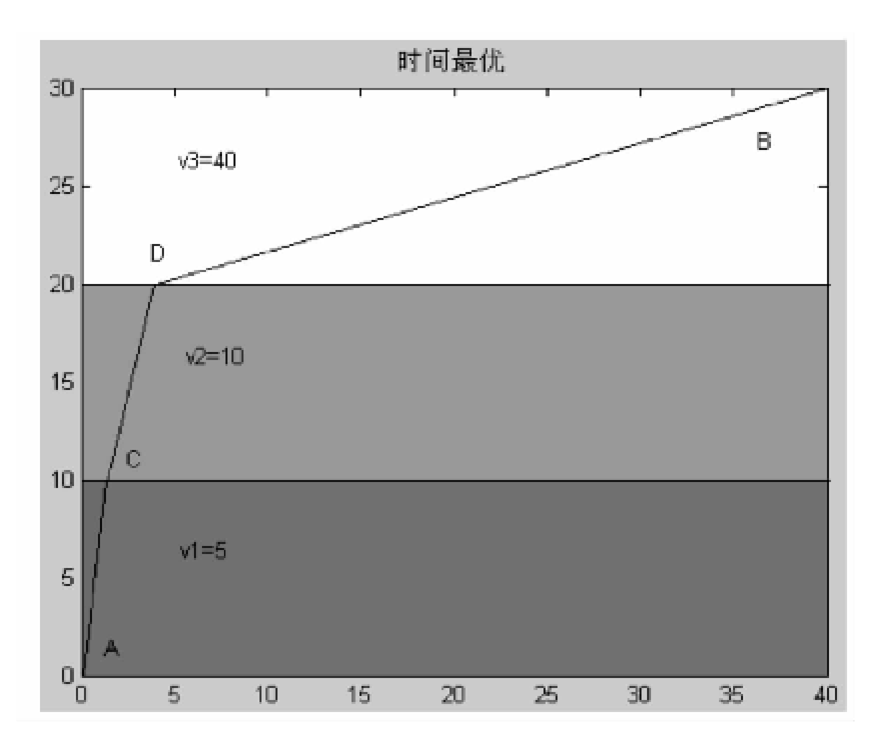


图 9-3 旅程中时间花费最小问题

由以上目标函数和约束条件即可使用 MATLAB 中非线性条件约束寻优函数 fmincon 进行计算。首先在 M 文件编辑器创建目标函数 timin 如下：

```
function y = timin(x)
y = 2* (1 + x(1)^2)^(1/2) + (1 + x(2)^2)^(1/2) + 0.25* (1 + x(3)^2)^(1/2);
```

接着在命令窗口输入如下程序：

```
≫x0 = [0;0;0];                                            %设置初值
≫Aeq = [1 1 1];beq = [4];                                 %输入等式约束
≫lb = [0,0,0];ub = [1;2;4];                               %设置上下限
≫options = optimset('largescale','off');                  %不使用大规模算法
≫[x,fval,exitflag,output] = fmincon(@timin,x0,[],[],Aeq,beq,lb,ub,[],options)
                                                          %调用寻优函数
Optimization terminated successfully:                     %最优化取得成功
First-order optimality measure less than options.TolFun and
   maximum constraint violation is less than options.TolCon
No active inequalities
x =                                                       %最优解
      0.1214
      0.2483
      3.6303
fval =                                                    %最优化的旅程时间,单位为小时
      3.9864
exitflag =
      1
```

```
output =
       iterations: 8
        funcCount: 44
         stepsize: 1
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: 9.6635e-007
     cgiterations: []
```

9.3 有约束最优化的图解

以上介绍的非线性最优化，是在找出问题的模型后，使用 MATLAB 的寻优函数，仔细地输入相关的数据后求解。如果稍不注意，未按寻优函数的语法要求格式输入，就会显示出错信息。本节介绍的图解法，则能显示最优化的几何意义，帮助理解解题过程。

图解分析法基本上可以分为三步，第一步为分析可行域，即根据约束条件，分析在 $x-y$ 平面上，应在那些区域内寻找目标函数的最优值。

第二步为画出目标函数的等值曲线族。

第三步则在可行域内，寻找等值曲线的最大或最小。下面举例于以说明。

【例 9-7】 已知目标函数 $z=\sqrt{(x-12)^2+(y-12)^2}$，约束条件为

$$x+y\leqslant 20;\ y\leqslant 14;\ x-3y\leqslant 0$$

求 z 最小值。

解：

根据约束条件，建立最优化的可行域，参阅图 9-4 的阴影面积。它是由 3 根直线和坐标轴 y 围成。这 3 根直线是

$$x+y=20;\ y=14;\ x-3y=0;$$

图中阴影面积是用着色函数 fill 绘成的。该命令如下

fill (x1, y1,'g')

式中，x1 为阴影四边形的横坐标向量x1 = [0 0 6 15 0]；y1 为阴影四边形的纵坐标向量y1 = [0 14 14 5 0]；g 表示充填绿色。

接着绘制目标函数的等值曲线。考虑目标函数 z^2 等于常数 C，则

$$y_1=12+\sqrt{C-(x-12)^2}\text{，用来绘制上半圆}$$

$$y_2=12-\sqrt{C-(x-12)^2}\text{，用来绘制下半圆}$$

绘制目标函数等值曲线的程序如下：

```
>>for C = 4:4:16                                 %目标函数值设置
      x = 12 - sqrt(C):0.01:12 + sqrt(C);        %设置 x 向量
      y1 = 12 + sqrt(C - (x - 12).^2);           %计算 y 向量
      y2 = 12 - sqrt(C - (x - 12).^2);
      plot(x,y1,x,y2);                           %绘制目标函数的等值线,参阅图 9-4
      hold on                                    %图形保持
  end
```

```
>>axis square                                  %设置坐标轴对称
>>axis([0,20,0,20])                            %设置坐标轴范围
```

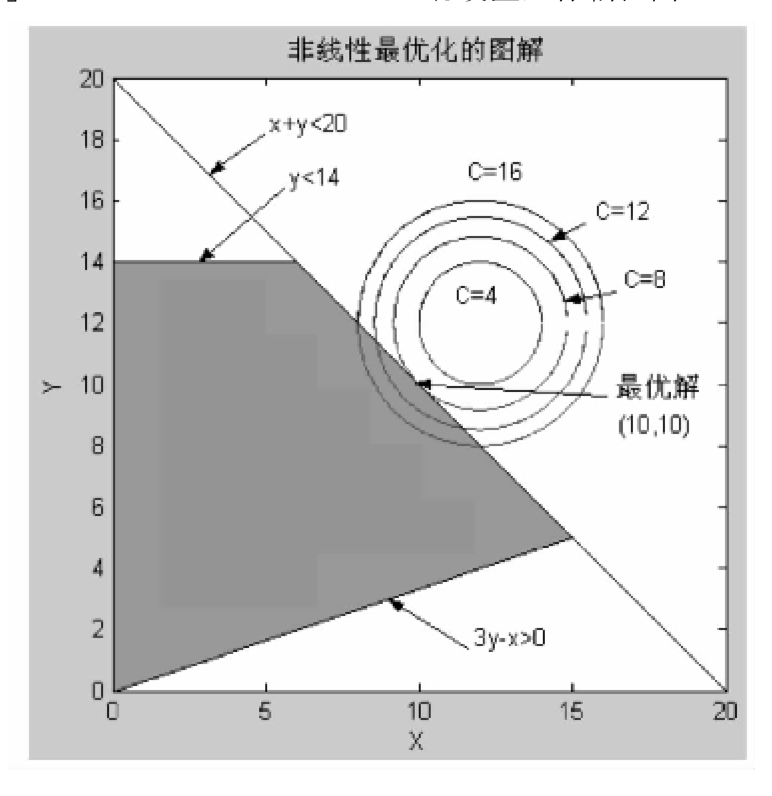


图 9-4 非线性最优化的图解

从图 9-4 可知目标函数的等值线 $C=8$，恰好与可行域相切。因此 $\min z=8$，即最优解为 8，相切点的坐标为（10，10）。

若用 fmincon 函数寻优则结果如下：

```
function z = optfun(x)                         %编制目标函数 M 文件
    z = ((x(1) - 12).^2 + (x(2) - 12).^2).^(1/2);
>>A = [1 1;1 -3;0 1]                           %输入不等式约束矩阵
A =

     1     1
     1    -3
     0     1

>>b = [20;0;14];                               %输入不等式约束矩阵的常数项向量
>>x0 = [0;0];                                  %设初值
>>options = optimset('largescale','off');      %选项,不执行大规模算法
>>lb = [5;5];ub = [20;20];                     %设置上、下限
>>[x,fval] = fmincon(@optfun,x0,A,b,[],[],lb,ub,[],options)
                                               %条件约束寻优
Optimization terminated successfully:          %最优化成功
First-order optimality measure less than options.TolFun and
  maximum constraint violation is less than options.TolCon
```

```
Active inequalities (to within options.TolCon = 1e-006):
  lower      upper      ineqlin      ineqnonlin
                            1
x =                                          %最优解
    10
10
fval =                                       %目标函数最优值与图解法 C^(1/2)相一致
    2.8284
```

9.4 二次规划

上一节提到的，具有约束条件的最优化中对目标函数的性质和阶次无限制，但对能否取得最优解是无法肯定的，要从解的结果来判别。但对于二次规划，若满足矩阵 H 的正定条件，它有惟一的最优解，二次规划的标准形式为

$$\min_{x} 1/2\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x}+\boldsymbol{f}^{\mathrm{T}}\boldsymbol{x}$$，它受约束于

线性不等式约束	$\boldsymbol{A}\boldsymbol{x}\leqslant\boldsymbol{b}$
线性等式约束	$\mathrm{Aeq}\boldsymbol{x}=\mathrm{beq}$
上、下边界	$\mathrm{lb}\leqslant\boldsymbol{x}\leqslant\mathrm{ub}$

式中，$\boldsymbol{H}$、$\boldsymbol{A}$、Aeq 为矩阵，$\boldsymbol{x}$、$\boldsymbol{f}$、$\boldsymbol{b}$、beq、lb、ub 为向量。

二次规划的书写形式为

```
x = quadprog(H,f,A,b)                                用于线性不等式约束
x = quadprog(H,f,A,b,Aeq,beq)                        用于等式兼不等式约束
x = guadprog(H,f,A,b,Aeq,beq,lb,ub)                  添加上,下边界
x = guadprog(H,f,A,b,Aeq,beq,lb,ub,x0)               添加初值
x = guadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)       添加选项
```

对于最优化输出有以下几种格式：

```
[x,fval]= quadprog(…)                                增加最优值输出
[x,fval,exitflag]= quadprog(…)                       增加退出标记
[x,fval,exitflag,outpot]= quadprog(…)                增加输出报告
[x,fval,exitflag,output,lambda]= quadprog(…)         增加拉格朗日乘子
```

【例 9-8】 求解如下二次规划问题

$$f(x_1,x_2,x_3)=\frac{1}{2}x_1^2+x_1x_2+x_1x_3+x_2^2+3x_2x_3+x_3^2-6x_1-7x_2-8x_3$$

受约束于

$$\begin{aligned}x_1+x_2+3x_3&\leqslant 5\\ x_1+2x_2+x_3&\leqslant 6\\ 3x_1+x_2+2x_3&\leqslant 13\end{aligned}$$

求 $\min(\boldsymbol{f})$。

解：

根据题设要求，写成矩阵形式如下：

$$\min\boldsymbol{f}(\boldsymbol{x})=1/2\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x}+\boldsymbol{f}^{\mathrm{T}}\boldsymbol{x}$$

其中 $\boldsymbol{H} = [1\ 1\ 1;1\ 2\ 3;1\ 3\ 2]$，$\boldsymbol{f} = [-6;-7;-8]$

$\boldsymbol{A} = [1\ 1\ 3;1\ 2\ 1;3\ 1\ 2]$，$\boldsymbol{b} = [5;6;13]$

```
>>H = [1 1 1;1 2 3;1 3 2];                                   %输入参数
>>f = [-6;-7;-8];
>>A = [1 1 3;1 2 1;3 1 2];
>>b = [5;6;13];
>>options = optimset('largescale','off');                    %不执行大规模算法
>>[x,fval] = quadprog(H,f',A,b,[],[],[],[],[],options)       %计算二次规划
Optimization terminated successfully.                        %最优化成功
    x =                                                      %最优解
    4.0000
    1.0000
         0
fval =                                                       %目标函数最优值
  -18.0000
```

9.5 线性最小二乘解

线性最小二乘解已经在第 3 章叙述过。若超定方程组 $\boldsymbol{Ax} = \boldsymbol{b}$，其中矩阵 $\boldsymbol{A}$ 为 $m \times n$ 矩阵，$\boldsymbol{b}$ 为 $m \times 1$ 向量，$\boldsymbol{x}$ 为 $n \times 1$ 向量，且 $m > n$，则 x 的最小二乘解可以用

左除法求解　　x = A \ b

或者用广义逆矩阵求解　　x = pinv(A) * b

本节将介绍可以用线性最小二乘解函数 lsqlin 来求解超定方程组即

x = lsqlin(A,b)

但是函数 lsqlin 增加了新的功能，它可以附有等式约束和不等式约束。函数 lsqlin 的目标函数为

$$\min_x \| Cx - d \|_2^2$$

受约束为

$$\boldsymbol{Ax} \leqslant \boldsymbol{b}$$
$$\mathrm{Aeq} * \boldsymbol{x} = \mathrm{beq}$$
$$\mathrm{lb} \leqslant \boldsymbol{x} \leqslant \mathrm{ub}$$

式中，$\boldsymbol{C}$、$\boldsymbol{A}$、Aeq 为矩阵，d、b、beq、lb、ub、x 为向量。

```
x = lsqlin(C,d,A,b,Aeq,beq)
x = lsqlin(C,d,A,b,Aeq,beq,lb,ub,x0)
x = lsqlin(C,d,A,b,Aeq,beq,lb,ub.x0,options)
[x,resnorm] = lsqlin(…)
[x,resnorm,residual] = lsqlin(…)
[x,resnorm,residual,exitflag] = lsqlin(…)
[x,resnorm,residual,exitflag,output] = lsqlin(…)
```

式中，x0 为初值，options 为选项，见 9.1 节。resnorm 为残差的范数，residual 为残差。

【例 9-9】 已知超定方程组 $\boldsymbol{Ax}=\boldsymbol{b}$，$\boldsymbol{A}=\begin{bmatrix}66 & 93 & 49\\73 & 91 & 35\\29 & 41 & 81\\40 & 89 & 57\end{bmatrix}$，$\boldsymbol{b}=[600\ 550\ 500\ 570]$，试用左除法，广义逆矩阵法和线性最小二乘法求解。

解：

```
>>A = [66 93 49;73 91 35;29 41 81;40 89 57]
A =                                 %输入矩阵 A
    66    93    49
    73    91    35
    29    41    81
    40    89    57
>>b = [600;550;500;570];            %输入常数项向量
>>x = A \ b                         %用左除法求解
x =
    1.8725
    3.0194
    3.9752
>>x1 = pinv(A)* b                   %用广义逆矩阵求解
x1 =
    1.8725
    3.0194
    3.9752
>>[x2,resnorm,residual,exitflag,output] = lsqlin(A,b)
x2 =                                %用线性最小二乘解求解
    1.8725
    3.0194
    3.9752
resnorm  =                          %残差范数
    1.0835
residual  =                         %残差
  -0.8244
    0.5911
    0.0923
    0.2145
exitflag =                          %退出标志,收敛
     1
output =                            %输出报告
        algorithm: 'slash'
```

```
    iteration: 0
firstorderopt: []
  cgiterations: []
```

计算结果表明，3 种算法都是相同的。

【例 9-10】 计算平面方程 $x_1+3x_2+5x_3=11$，求平面到原点的垂直距离及垂足的坐标。

解：

平面方程式到原点的距离为

$$\min_x z = x_1^2 + x_2^2 + x_3^2$$

约束条件为

$$x_1+3x_2+5x_3=11$$

由此得

```
>>C = eye(3)                                                  %输入矩阵 C
C =
    1    0    0
    0    1    0
    0    0    1
>>d = zeros(3,1);                                             %输入向量 d
>>A = []; b = [];                                             %无不等式约束
>>Aeq = [1 3 5]; beq = 11;                                    %等式约束向量
>>options = optimset('largescale','off');                     %不执行大规模算法
>>[x,fval] = lsqlin(C,d,A,b,Aeq,beq,[],[],[],options)         %调用线性最小二乘法
Optimization terminated successfully.
x =                                                           %最优解
    0.3143
    0.9429
    1.5714
fval =                                                        %目标值,即原点到平面的距离
    3.4571
```

第 9 章习题

9-1 直角坐标平面上有两点 $A(x_1,y_1)$，$B(x_2,y_2)$，求在 x 轴上找一点 $(0,x)$，使 $L=AC+BC$ 最小。

9-2 在椭圆方程 $\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$ 中嵌入一矩形，矩形的边，平行于坐标轴，求最大矩形的面积。

9-3 一个金属薄壁的瓶为圆柱形，直径为 D，高为 H，上端为半球形。若瓶的材料面积为 S，在 S 一定时，D 和 H 应如何配置才能使用 V 最大。

9-4 一张 30cm × 50cm 的镀锌铁皮，拟剪去四角，弯成矩形容器，问应剪去多少，才使容积最大。

9-5 轮船在航行中的燃料消耗 $P=a+bV^3$，a、b 为常数，V 为航行速度。为使燃料消耗最小，求最优航行速度。

9-6　设圆桌面的半径为 r，桌面上正中央高为 H 处安放一吊灯，为使桌子边缘的照度最大，求 H。已知照度公式为

$$I = k\,\frac{\sin a}{L^2}$$

式中，k 为光源强度，a 为光线与桌面的倾斜角，L 为光源与被照点的距离。

第 10 章　MATLAB 在线性规划中的应用

线性规划（Linear Programming）是广泛应用于生产计划、运输计划、分配计划和经营计划的编制。线性规划又称为线性最优化（Linear Optimization），因此，它是最优化的一个分支。线性规划首先要有一个目标函数，如产品成本、能源消耗、产量或利润等，作为一个衡量计划优劣的依据。其次，线性规划需要有一组控制变量或称为决策变量，控制变量能人为加以改变的，而目标函数必须是控制变量的线性组合，如果不是线性组合，则是属于非线性规划的范畴。第三线性规划需要一组约束条件。约束条件可以是等式约束条件，也可以是不等式约束条件，也可以二者兼有。约束条件主要根据市场供求情况，企业设备情况和劳动力情况决定的。线性规划的目标是在控制变量受线性等式约束或线性不等式约束的条件下，以及二者兼有的条件下使目标函数达到最大或最小。

线性规划是目前应用数学中最有成效的学科之一。它的主要用途有以下几方面。

1. 编制计划，使运输费用最小

例如某公司在全国有若干个生产基地，它生产的产品，需要供应各地销售商。生产基地到各地销售商的运费和供应量是不同的。如何设计一个分配、运输方案，既满足产销平衡，又使运费最小。

2. 编制作业计划，使运输中的吨•千米（t•km）最小

某网络公司，每天根据订单发送产品。如何设计出送货计划，使送货司机所走的路程最短，或运输中的吨•千米最小。

3. 设计原料的配比使成本最低

将各种原料混合以达到一定需求。由于原料的价格和所占的需求成分不同，如何设计出一种配方，既满足混合物成分的要求，又实现成本最低。

4. 编制生产计划，使企业利用润最大。

假若某企业生产 A、B、C3 种产品，每种产品的利润为 Pa、Pb、Pc。每种产品需经过 r、s、t 的加工工序，由于 r、s、t 工序的加工能力受到限制，求如何编制生产计划使企业利润最大。

5. 最优下料

在产品加工中，需从原材料下料，如何使余下边脚料最小，使原材料最大限度的利用，也是线性规划的任务。

6. 最小贮存

为了生产的连续进行，不能因缺少原材料而停产，所以仓库是必不可少的。但过于大的仓库和原材料的贮备是增加企业的负担。如何使企业在生产效率达到最大的同时，实现企业的原材料和半成品贮存达到最小，也是企业的重要任务。

7. 人力资源的合理使用

人事管理人员根据员工的特长，任务的性质、难度和进度要求，合理地编制人员组合，以最短的工期完成任务。

8. 招、投标工作中报价分析

对卖方的报价单进行分析，对价格、预期质量、完工日期、资历和信誉度进行多目标的评估。帮助买方实现选取投标单位的最优化。

上述举例远远不足于概括线性规划在实际上应用。所以目前有关线性规划的计算机软件十分丰富。下面列举有关线性规划的计算机软件。有的软件能处理 30000 个决策变量和 5000 个等式或不等式约束条件。

(1) Excel97 以上版本中有关规划求解的子菜单

(2) MATLAB

(3) Lingo

(4) Gams

(5) Cplex

(6) QSB

有关软件信息可以从软件名称的相应网站找到。有些属于学习的版本，还可以免费下载。线性规划的难点在于寻找问题的线性规划数学模型。如果你能正确地写出数学模型，那么你的任务已经完成了 80%，剩下的问题是利用计算机软件解题和分析结果了。

关于列出问题的数学模型，很重要的一环是确定目标函数。目标函数应该是企业的关键指标。例如利润、成本、能源消耗、劳动生产率等。其次，目标函数应该是可计量的。例如企业透明度、廉政指数、职工素质指数、健康度、营养度等指标难以计量，则不宜作目标函数。再次考虑目标函数与哪几个决策变量有关，以便列出目标函数的方程式。有时目标函数不止一个，这时需要采用多目标寻优，把多目标综合起来，对每个目标值添加权因子，使综合目标值达到最优，而对每一个单独的目标，就不是最优。

确定决策变量也是确定数学模型的重要因素。决策变量应该是可以控制的，对目标函数有影响，影响的程度可以从已知数据通过数学推理取得，也可以从试验中取得。

确定等式和不等式的约束条件，也是不可缺少的。因为决策变量的可控范围一般受到限制。例如设备条件限制，加工时间限制，原材料供应量的限制，劳动力的限制等。其中最基本的限制为在大于零的限制。因为产量、成本、能源消耗等变量不可能为负数。

线性规划常用的方法是单纯形法，这个方法是 1947 年由乔奇•旦泽（George Dantzig）创造的，它最初用于空军计划的编制。单纯形法是一种迭代求解法。该方法从某一个基本可行解（凸集的某个顶点）开始，计算该点是否是极值。若不是，则构造新的基本可行解，（它是凸集的相邻顶点）使目标值得到改进，经有限步迭代，则得到最优解。由于计算机特别擅长于迭代运算，所以到 20 世纪 70 年代就出现用计算机解算线性规划问题，用不着用单纯形计算表来解算了。由于个人计算机的诞生，更使得单纯形法如虎添翼。

10.1 线性规划的图解法

对于决策变量不超过 2～3 个，则线性规划可以用图解法来解决。用图解法主要是为了便于了解线性规划的用途。为了说明线性规划的图解法，这里举一个例子。

【例 10-1】 某工厂生产两种产品 A 和 B。产品 A 的利润每台150元，产品 B 的每台利润为 200 元。产品 A、B 每台所需加工工时及设备所能提供加工工时见表 10-1。问如何安排生产才能使工厂的利润最大。

表 10-1　某厂生产产品的工时及工序

工时 产品 / 工序	A	B	每月可用工时
机械加工	3.0	2.0	1500
装　　配	2.0	3.0	1500

解：

第一步，列出问题的数学模型

设 A 产品的生产台数为 N_a，B 产品的生产台数为 N_b，总利润为 P。则目标函数为求最大值。

$$\max P = 150N_a + 200N_b$$

约束条件为

$$2N_a + 3N_b \leqslant 1500$$

$$3N_a + 2N_b \leqslant 1500$$

$$N_a，\ N_b \geqslant 0$$

第二步，对不等式约束条件进行图解

得出可行解范围如图 10-1 所示阴影。对上面的问题，用横坐标 X 表示产品 N_a，用纵坐标 Y 表示产品 N_b。把两个不等式约束条件，绘制成直线，则在阴影线内，表示符合约束条件的解的集合。

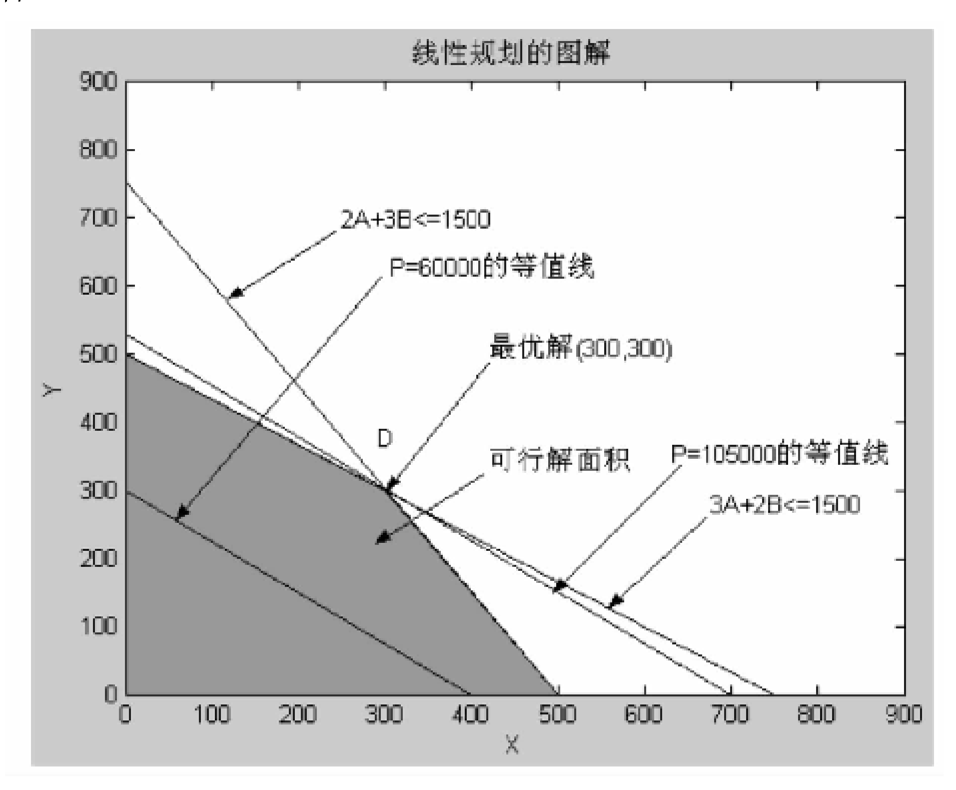


图 10-1　简单的线性规划问题的图解法

第三步，在可行解区域内绘制目标函数等值线

在本例中画出 $P = 60000$ 的等值线，这等值线在 X 轴上的截距为 $N_a = 60000/150 = 400$，在 Y 轴上的截距为 $N_b = 60000/200 = 300$。连接这两点的直线，即为目标函数为 60000 的等

值线。在这线上每一点，它符合约束条件，同时它对利润的贡献值为 60000。

第四步，寻找最优解

作平行于目标函数等值线的线，使目标值不断扩大。过 D 点作平行于目标函数的等值线，则该线为目标函数的最大值。因为再向上移动，目标值 P 虽然增大，但已超出约束条件的范围。D 点的坐标可从解下列方程式得到。

$$2N_a + 3N_b = 1500$$

$$3N_a + 2N_b = 1500$$

则得最优解 $$N_a = N_b = 300$$

目标函数最大值 $$\max P = 150N_a + 200N_b = 105000$$

因此应该这样安排生产，A、B 两种产品各生产 300 台。利润为 1050000 元。

10.2 线性规划问题的 MATLAB 解法

MATLAB 解线性规划的函数为 linprog。它对于目标函数，等式和不等式的约束条件的表述如下：

使目标函数到最小 $$\min_x \boldsymbol{f}^T\boldsymbol{x}$$

该式受不等式约束于 $$\boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b}$$

和等式约束 $$\mathrm{Aeq}\boldsymbol{x} = \mathrm{beq}$$

以及上、下界限制 $$\mathrm{lb} \leqslant \boldsymbol{x} \leqslant \mathrm{ub}$$

这里 $\boldsymbol{f}$、$\boldsymbol{x}$、$\boldsymbol{b}$、beq、lb、ub 是向量，而 $\boldsymbol{A}$、Aeq 为矩阵。式中 $\boldsymbol{f}$ 为目标函数的系数向量，$\boldsymbol{x}$ 为决策变量。$\boldsymbol{b}$ 为不等式约束表达式右端的常数向量。beq 为等式约束表达式右端的常数向量。lb、ub 为 x 的下界和上界。如果求解目标函数的最大值，则可将目标函数改为负值，把目标函数改为求解最小值。解出后再把目标函数的符号改回来。若不等式约束是大于等于形式，则可以在不等式两端乘上负号，把不等式的方向改过来。

求解线性规划的函数 linprog 的书写格式如下：

最优解 x = linprog(f, A, b)

x = linprog(f, A, b, Aeq, beq, lb, ub)

x = linprog(f, A, b, Aeq, beq, lb, ub, x0)

x = linprog(f, A, b, Aeq, beq, lb, ub, x0, option)

最优解及附有输出值的形式为 [x, fval] = linprog(…)

[x, fval, exitflag] = linprog(…)

[x, fval, exitflag, output] = linprog(…)

[x, fval, exitflag, output, lambda] = linprog(…)

x = linprog (f, A, b) 用于不等式约束，使目标函数 $\min \boldsymbol{f}^T\boldsymbol{x}$ 为最小的解。

x = linprog(f, A, b, Aeq, beq, lb, ub)用于具有等式约束和不等式约束，使目标函数为最小的解。若只有等式约束，则不等式约束的矩阵 A 和向量 b 需用空阵 [] 代替。

x = linprog(f, A, b, Aeq, beq, lb, ub, x0)与上式相同，但增加了初值设置，这种设置仅对中规模算法有效。

x = linprog(f, A, b, Aeq, beq, lb, ub, x0, options)在这个式子中指定了结构选项，这选项由

optimset（最优化设置）来设置参数。

对于算式[x,fval] = linprog(…)则返回最优解 x 的同时，附加目标值 fval。

对于算式[x,fval,exitflag] = linprog(…)返回一个 exitflag（退出标记）值，它描述退出运算的条件。

对于算式[x,fval,exitflag,output] = linprog(…)返回一个结构输出，它包含关于最优化解算的信息。

对于算式[x,fval,exitflag,output,lambda] = linprog(…)返回一个 lambda 结构参数，它表示了拉格朗日乘子内容。

【例 10-2】 承接上例【例10-1】用 MATLAB 求解。

解：

```
>>A = [2 3;3 2]                          %输入不等式约束矩阵
A =
      2     3
      3     2
>>b = [1500;1500];                       %输入不等式约束,右侧常数向量
>>f = [-150;-200];                       %由于求目标函数的极大,故将原目标函数改为负值,
                                          而求极小
>>[x,fval] = linprog(f,A,b)              %求解线性规划
Optimization terminated successfully.    %最优化成功
x =                                      %最优解
   300.0000
   300.0000
fval =                                   %最优解的目标函数值，再取负值，即为最大利润，
                                          与上例图解法的结果相一致
  -1.0500e+005
```

【例 10-3】 考虑混合饲料配比，它由玉米粉和大豆饼组成。其配比要求和价格见表10-2。求最优配比使饲料成本最低。

表 10-2 某混合饲料的配比要求和价格

品名 / 项目	玉 米	大 豆 饼	配 比 要 求
发热量	4Mcal[①]/kg	2Mcal/kg	大于 2.8Mcal/kg
蛋白含量	90g/kg	300g/kg	大于 220g/kg
价格	2 元/kg	1.6 元/kg	

① Mcal 为兆卡，1Mcal/kg = 4.1868 × 10^6J/kg。

解：

设混合饲料中玉米每公斤含量为 $x(1)$，大豆饼含量为 $x(2)$，则目标函数为

求最小 $$\min Z = 2x(1) + 1.6x(2)$$

约束条件为

$$4x(1) + 2x(2) \geqslant 2.8$$

$90x(1)+300x(2)\geqslant 220$

$x(1)+x(2)=1$

$x(1)、x(2)\geqslant 0$

在 MATLAB 命令窗口输入如下程序：

```
≫A = [4 2;90 300];                          %输入不等式约束矩阵
≫b = [2.8;220];                             %输入常数项向量
≫f = [2 ;1.6];                              %输入目标函数
≫lb = zeros(2,1);                           %输入下限约束
≫Aeq = [1 1];beq = 1                        %输入等式约束
≫A = - A;b = - b;                           %将约束不等式改写成小于等于
≫[x,fval] = linprog(f,A,b,Aeq,beq,lb)       %调用计算线性规划函数
Optimization terminated successfully.       %最优解成功
x =                                         %最优解,配比为 0.4(玉米):0.6(豆饼)
    0.4000
    0.6000
fval =                                      %饲料每公斤最低成本
    1.7600
```

【例 10-4】 超市收银员的最优设置。考虑超市的收银员周一至周四需要人数为10人。但周五至周日因顾客增加，周五需要收银员 12 人，周六及周日需要收银员 14 人。问应如何配备最少的收银员来满足上述要求。收银员工作日，为每周 5 天。

解：

设周一开始上班的收银员人数为 $x(1)$，简称 1 班

设周二开始上班的收银员人数为 $x(2)$，简称 2 班

…

设周日开始上班的收银员人数为 $x(7)$，简称 7 班，则目标函数为

$$\min Z = x(1)+x(2)+x(3)+\cdots+x(7)$$

目标函数系数

$$f=[1\quad 1\quad 1\quad 1\quad 1\quad 1\quad 1]$$

约束条件为

$$x(4)+x(5)+x(6)+x(7)+x(1)=10$$
$$x(2)+x(5)+x(6)+x(7)+x(1)=10$$
$$x(3)+x(2)+x(6)+x(7)+x(1)=10$$
$$x(4)+x(3)+x(2)+x(7)+x(1)=10$$
$$x(5)+x(4)+x(3)+x(2)+x(1)=12$$
$$x(6)+x(5)+x(4)+x(3)+x(2)=14$$
$$x(7)+x(6)+x(5)+x(4)+x(3)=14$$
$$x(1),x(2),\cdots,x(7)\geqslant 0$$

在 MATLAB 命令窗口输入如下程序：

```
≫clear                                      %清工作空间
≫f = ones(7,1);                             %设目标函数向量
≫Aeq = [1 0 0 1 1 1 1;1 1 0 0 1 1 1;1 1 1 0 0 1 1;1 1 1 1 0 0 1;1 1 1 1 1 0 0;0 1 1 1 1 1 0;
```

```
        0 0 1 1 1 1 1]                          %输入等式约束矩阵
Aeq =
     1   0   0   1   1   1   1
     1   1   0   0   1   1   1
     1   1   1   0   0   1   1
     1   1   1   1   0   0   1
     1   1   1   1   1   0   0
     0   1   1   1   1   1   0
     0   0   1   1   1   1   1
>>beq = [10;10;10;10;12;14;14];                  %输入等式约束矩阵右侧常数项
>>lb = zeros(7,1);                               %大于等于零约束
>>[x,fval] = linprog(f,[],[],Aeq,beq,lb)         %调用线性规划计算函数
     Optimization terminated successfully.       %最优化成功
x =                                              %最优解
     0.0000
     2.0000
     4.0000
     2.0000
     4.0000
     2.0000
     2.0000
fval =                                           %最少收银员人数为 16 人
     16.0000
```

收银员班次安排如下：

星期一，1 班无人，由前期上班的 4、5、6、7 班担任。共 10 位收银员；

星期二，2 班 2 人，再加 5、6、7 班。共 10 位收银员；

星期三，3 班 4 人，再加 2、6、7 班。共 10 位收银员；

星期四，4 班 2 人，再加 2、3、7 班。共 10 位收银员；

星期五，5 班 4 人，再加 2、3、4 班。共 12 位收银员；

星期六，6 班 2 人，再加 2、3、4、5 班。共 14 位收银员；

星期日，7 班 2 人，再加 3、4、5、6 班。共 14 位收银员。

由于本题凑巧，答案恰好为整数。若出现非整数的情况，可以试用 10.6 节的整数规划。

10.3 运输问题

运输问题在线性规划中得到广泛的应用。运输问题的一般描述为某产品有生产地 $A_1 \sim A_n$，产量分别为 $Q_1 \sim Q_n$，另有 m 个销售地 $B_1 \sim B_m$，销售地的需求量分别为 $S_1 \sim S_m$，生产地 A_i 到销售地 B_j 的单位运费为 C_{ij}，运输量为 Q_{ij}，则目标函数为总运费

$$\min P = \sum_{i=1}^{n} \sum_{j=1}^{m} C_{ij} Q_{ij}$$

产地的约束条件为

$Q_{11}+Q_{12}+\cdots+Q_{1m}\leqslant Q_1$

$Q_{21}+Q_{22}+\cdots+Q_{2m}\leqslant Q_2$

$\vdots$

$Q_{n1}+Q_{n2}+\cdots+Q_{nm}\leqslant Q_n$

运输量与需求量的平衡条件为

$Q_{11}+Q_{21}+\cdots+Q_{n1}=S_1$

$Q_{12}+Q_{22}+\cdots+Q_{n2}=S_2$

$\vdots$

$Q_{1m}+Q_{2m}+\cdots+Q_{nm}=S_m$

$Q_{ij}\geqslant 0\ (i=1:n,\ j=1:m)$

由此目标函数和约束条件，即可使用线性规划来求最优解。

【例 10-5】 设有四个城市 A、B、C、D。它的粮食分别由三个产粮地 X、Y、Z 供给。这四个城市的人口和年粮食需求量见表 10-3，而 X、Y、Z 运粮到 A、B、C、D 的每吨运费见表 10-4。产粮地 X、Y、Z 的粮食产量见表 10-5。

表 10-3 某四个城市的人口和年粮食需求量

城　　市	A	B	C	D
人口/万人	23.3	18	24.4	30.5
年粮食需求/万吨	4.2	3.3	4.5	5.8

表 10-4 三个产粮地运到四个城市的运费 （单位：元）

城市 产 粮 地	A	B	C	D
X	32	32	65	75
Y	52	60	110	70
Z	25	80	30	40

表 10-5 三个产粮地的粮食产量

产　粮　地	X	Y	Z
粮食产量/万吨	2.5	8.2	7.3

问题的要求为：满足每个城市的粮食需要量，并使总运费达到最小。

解：

(1) 随意调配时粮食运费

如果不采用线性规划，而采用随意调配则运费计算如下：

- 将 X 地的 2.5 万吨和 Y 地的 1.7 万吨调到 A 城，则运费为

$$P1=(2.5\times 32+1.7\times 52)\text{万}=168.4\text{万}$$

- 将 Y 地的 3.3 万吨调到 B 城，则运费为

$$P2=(3.3\times 60)\text{万}=198\text{万}$$

- 将 Y 地剩下的 3.2 万吨，和 Z 地的 1.3 万吨调到 C 城，则运费为

$$P3 = (3.2 \times 110 + 1.3 \times 30)\text{万} = 391\text{万}$$

- 将 Z 地的 5.8 万吨调入 D 城，Z 地剩下粮食为 0.2 万吨，则运费为

$$P4 = (5.8 \times 40)\text{万} = 232\text{万}$$

故总运费为

$$P = P1 + P2 + P3 + P4 = (168.4 + 198 + 391 + 232)\text{万} = 989.4\text{万}$$

(2) 设定约束条件和目标函数

设粮食产地与需粮城市之间运输量为 Q_{ij}，见表 10-6。

表 10-6 粮食产地与需粮城市间运输量

城市 / 产粮地	A	B	C	D
X	Q11	Q12	Q13	Q14
Y	Q21	Q22	Q23	Q24
Z	Q31	Q32	Q33	Q34

由此可列出约束方程式如下：

$$Q_{11} + Q_{12} + Q_{13} + Q_{14} \leqslant 2.5 \tag{10-1}$$

$$Q_{21} + Q_{22} + Q_{23} + Q_{24} \leqslant 8.2 \tag{10-2}$$

$$Q_{31} + Q_{32} + Q_{33} + Q_{34} \leqslant 7.3 \tag{10-3}$$

$$Q_{11} + Q_{21} + Q_{31} = 4.2 \tag{10-4}$$

$$Q_{12} + Q_{22} + Q_{32} = 3.3 \tag{10-5}$$

$$Q_{13} + Q_{23} + Q_{33} = 4.5 \tag{10-6}$$

$$Q_{14} + Q_{24} + Q_{34} = 5.8 \tag{10-7}$$

$$Q_{ij} \geqslant 0$$

其中 i 代表产量地，j 代表需粮城市。

为使总的运输费用最小，这就要求运输费用的目标函数 P 为最小，即

$$\min P = C_{11}Q_{11} + C_{12}Q_{12} + C_{13}Q_{13} + C_{14}Q_{14} + C_{21}Q_{21} + C_{22}Q_{22} + C_{23}Q_{23} + C_{24}Q_{24} + C_{31}Q_{31} + C_{32}Q_{32} + C_{33}Q_{33} + C_{34}Q_{34} \tag{10-8}$$

其中决策变量为 12 个，即 $Q_{11} \sim Q_{14}$；$Q_{21} \sim Q_{24}$；$Q_{31} \sim Q_{34}$

其中 C_{ij} 为运输费单价，见表 10-4 对应数字。

(3) 规划求解

在 MATLAB 命令窗口输入如下程序：

```
≫f = [32;32;65;75;52;60;110;70;25;80;30;40];    %根据表 10-4 和式（10-8）输入目标函数
≫v = ones(1,4)                                  %设置向量 v，z 为设置约束矩阵 A 做准备
v =
    1    1    1    1
≫z = zeros(1,4)
z =
```

```
0       0       0       0
>>A = [v, z, z; z, v, z; z, z, v]                 % 根据式(10-1) ~ 式(10-3),输入不等式
                                                    约束矩阵 A

A =
     1   1   1   1   0   0   0   0   0   0   0   0
     0   0   0   0   1   1   1   1   0   0   0   0
     0   0   0   0   0   0   0   1   1   1   1   1
>>b = [2.5;8.2;7.3];                               % 由表 10-5 输入不等式约束常数向量
>>v1 = eye(4)                                      % 设置幺矩阵,为设置等式约束矩阵
                                                    Aeq 做准备

v1 =
     1   0   0   0
     0   1   0   0
     0   0   1   0
     0   0   0   1
>>Aeq = [v1, v1, v1]                               % 根据式(10-4) ~ 式(10-7)输入等式约
                                                    束矩 Aeq

Aeq =
     1   0   0   0   1   0   0   0   1   0   0   0
     0   1   0   0   0   1   0   0   0   1   0   0
     0   0   1   0   0   0   1   0   0   0   1   0
     0   0   0   1   0   0   0   1   0   0   0   1
     >>beq = [4.2;3.3;4.5;5.8];                    % 根据表 10-3 输入等式约束常数项
>>lb = zeros(12, 1);                               % 设置决策变量下限
>>[x, fval] = linprog(f, A, b, Aeq, beq, lb)       % 线性规划计算
Optimization terminated successfully.              % 最优化成功
x =                                                % 最优解
     0.0000
     2.5000
     0.0000
     0.0000
     4.2000
     0.8000
     0.0000
     3.0000
     0.0000
     0.0000
     4.5000
     2.8000
```

fval =   %最优解时的目标值

803.4000

(4) 结论

由运算结果表明，经线性规划后总运费为 803.4 万元，比随意调配的总运费 989.4 万元减少了 186 万元。山决策变量最优解，可得产粮地运送粮食数字见表 10-7。山此可见，采用线性规划后，总运费达到最小，所以满足运输最优化。

表 10-7　由决策变量最优解所得运粮数字　　（单位：万吨）

城市 / 产粮地	A	B	C	D
X	0	2.5	0	0
Y	4.2	0.8	0	3
Z	0	0	4.5	2.8

10.4　最大利润问题

在工厂编制生产计划中，使产品的计划利润最大是通常的目标。本章【例 10-1】就是简单的求最大利润问题。在这个例子中，由于各个产品的加工工时不同，因此在加工工时的约束条件下，尽可能多的生产价格高的产品，而不管这种产品是否能销售出去。另一种最大利润表述方法是由于产品中所包含的零部件不同，而零部件的生产能力又受到约束的情况下如何求最大利润。

【例 10-6】　某工厂生产 A、B、C 三种产品，产品每台利润分别为600、500 和 400 元。它所用部件 P1 ~ P4 和部件的生产能力见表 10-8。求如何安排 A、B 和 C 的生产计划，使产品的计划利润最大。

表 10-8　某产品所用部件及其部件的生产能力

部件 / 产品	P1/件	P2/件	P3/件	P4/件	产品每台计划利润/元
A	2	1	1	1	600
B	1	2	1	2	500
C	1	1	2	0	400
部件每月生产能力/件	1000	800	800	750	—

解：

设 A、B、C 三种产品每月计划生产数量为 x_1、x_2、x_3 台，则计划利润 Z 求最大为

$$\max Z = 600x_1 + 500x_2 + 400x_3$$

它的目标函数向量为　$\boldsymbol{f} = [-600; -500; -400]$，它的约束条件为

$$2x_1 + x_2 + x_3 \leqslant 1000$$

$$x_1 + 2x_2 + x_3 \leqslant 800$$

$$x_1 + x_2 + 2x_3 \leqslant 800$$

$$x_1 + 2x_2 \leqslant 750$$

$$x_1、x_2、x_3 \geqslant 0$$

在 MATLAB 命令窗口输入如下程序:

```
>>f = [-600  -500  -400];                      % 目标函数向量,由于求极小,故
                                               取负号
>>A = [2 1 1;1 2 1;1 1 2;1 2 0]                % 不等式约束矩阵
A =
    2   1   1
    1   2   1
    1   1   2
    1   2   0
>>b = [1000;800;800;750];                      % 不等式约束矩阵右侧常数项向量
    >>lb = zeros(3,1);                         % 决策变量的下限
>>Aeq = [];beq = [];                           % 等式约束是空矩阵
>>[x,fval] = linprog(f,A,b,Aeq,beq,lb)         % 线性规划计算
Optimization terminated successfully.          % 最优化成功
x =                                            % 最优解,产品 A 生产 350 台,B、C 各生
                                               产 150 台
  350.0000
  150.0000
  150.0000
fval =                                         % 最大计划利润取负的 fval 为 345000 元
  -3.4500e+005
>>A* x                                         % 核对约束条件是否满足
ans =
  1000.0000
   800.0000
   800.0000
   650.0000
>>f'* x                                        % 核对目标函数
ans =
  -3.4500e+005
```

【例 10-7】 某化工厂用三种原料 A、B、C，生产甲、乙、丙三种产品。每项产品的原料需用量和实际可供应量以及每项产品的利润见表 10-9，试制订计划，使每日的利润最大。

表 10-9 某化工厂生产三项产品的原料需用量和实际可供应量及每项产品的利润

原料 \ 品种	甲/吨	乙/吨	丙/吨	原料每日可供应量/千克
A	2	3	0	1500
B	0	2	4	800
C	3	2	5	2200
每吨利润/百元	3	4	10	—

解：

```
≫A = [2 3 0;0 2 4;3 2 5]                %设置不等式约束矩阵
A =
     2   3   0
     0   2   4
     3   2   5
≫b = [1500;800;2200];                  %设置约束值,即每日原料可供应量
≫f = [-3;-4;-10];                      %设置目标函数,取负号是为了求最大值
≫lb = zeros(3,1);                       %设置控制变量的下限
≫x = linprog(f,A,b,[],[],lb)            %解线性规划
Optimization terminated successfully.   %最优化成功
x =                                     %最优解
  400.0000
    0.0000
  200.0000
≫profit = -f'*x                         %最大利润
profit =
  3.2000e+003
≫res = b-A*x                            %原料冗余
res =
  700.0000                              %原料 A 冗余 700kg
    0.0000                              %原料 B、C 用完
    0.0000
```

10.5 最小成本问题

如何降低产品成本是企业所关心的问题。降低成本可以从节约原材料，减小能源消耗，提高劳动生产力方面着手。线性规划对于节约下料，符合成分要求的最低成本配方等，颇有建树。下面举例说明。

【例 10-8】 某家具厂生产折叠椅，每只折叠椅需用1.5m 钢管 1 根，0.9、0.8、0.6m 钢管各两根。已知进货钢管每根长 2.5m。问钢管应如何下料，才使用料最省。

解：

解法一：

设钢管的下料方案为 1～8，每种方案的下料根数分别为 $x_1 \sim x_8$，每种规格的钢管需用量见表 10-10。

设置目标函数 Z 为用料总根数，并使之最小，则

$$\min Z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8$$

等式约束矩阵

$$Aeq = [1\ 1\ 0\ 0\ 0;1\ 0\ 2\ 1\ 1;0\ 1\ 0\ 1\ 2;0\ 0\ 1\ 1\ 0]$$

常数项向量

$$beq = [100;200;200;200]$$

表 10-10 某家具厂生产折叠椅的下料方案及每种规格的钢管需用量

截法 规格	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	需用量
1.5	1	1	1	0	0	0	0	0	100
0.9	1	0	0	1	1	1	0	0	200
0.8	0	1	0	1	2	0	2	1	200
0.6	0	0	1	1	0	2	1	2	200
余料	0.1	0.2	0.4	0.2	0	0.4	0.3	0.5	—

在 MATLAB 命令窗口输入如下程序：

```
>>Aeq = [1 1 1 0 0 0 0 0;1 0 0 1 1 1 0 0;0 1 0 1 2 0 2 1;    %根据表 10-10 输入等式约束
      0 0 1 1 0 2 1 2]                                      矩阵
Aeq =
     1   1   1   0   0   0   0   0
     1   0   0   1   1   1   0   0
     0   1   0   1   2   0   2   1
     0   0   1   1   0   2   1   2
>>beq = [100:200;200;200];                                  %输入常数项向量
>>lb = zeros(8,1);                                          %输入下限
>>f = ones(8,1);                                            %输入目标函数向量
>>[x,fval] = linprog(f,[],[],Aeq,beq,lb)                    %线性规划计算
Optimization terminated successfully.                       %最优化成功
x =                                                         %最优解
     57.1429
     21.4286
     21.4286
     47.6190
     47.6190
     47.6190
     11.9048
     11.9048
fval =                                                      %最优解时的目标函数
   266.6667
>>x1 = ceil(x)'                                             %取整,行向排列
x1 =
     58   22   22   48   48   48   12   12
>>Aeq* x1
ans =                                                       %各种钢管规格的根数
     102
     202
```

```
    202
    202
≫sum(x1)                                    %所需钢管总数
ans =
    270
```

解法二：

有时采用等值约束时会找不到最优解。这时可改用不等式约束求解，即

$$\boldsymbol{Ax} \geqslant \boldsymbol{b}$$

由于求解线性规划的标准形式是小于等于，所以将不等式二端乘负，则不等式约束改为

$$-\boldsymbol{Ax} \leqslant -\boldsymbol{b}$$

对于本例，则 $\boldsymbol{A} = -\mathrm{Aeq}$，$\boldsymbol{b} = -\mathrm{beq}$，则线性规划的解为

[x1,fval] = linprog(f,A,b,[],[],lb)

```
≫A = -Aeq;b = -beq;                         %将等式约束改为不等式约束,其他
                                             参数不变
≫f = ones(8,1);
≫[x1,fval1] = linprog(f,A,b,[],[],lb);      %线性规划求解
Optimization terminated successfully.
≫x1'
ans =                                       %最优解与等式约束相近
    55.4101   22.2950   22.2950   48.1966   48.1966   48.1966   11.0384   11.0384
≫fval1
fval =                                      %目标函数则与等式约束的目标函
                                             数相同
  266.6667
```

解法三：

如果考虑目标函数为钢管截下的余料最少，则目标函数应为

$$\min \boldsymbol{Z} = \boldsymbol{f}^T\boldsymbol{x} = [0.1\ 0.2\ 0.4\ 0.2\ 0\ 0.4\ 0.3\ 0.5]\boldsymbol{x}$$

而约束条件为 Aeq* $\boldsymbol{x}$ = beq，则最优解为

```
    ≫f = [0.1;0.2;0.4;0.2;0;0.4;0.3;0.5];    %输入目标函数
≫[x2,fval2] = linprog(f,[],[],Aeq,beq,lb);  %规划求解
Optimization terminated successfully.
≫x2'                                        %最优解
ans =
    57.9855   20.4177   21.5968   47.3021   48.0852   46.6271   11.4576   13.1947
≫fval2
fval2 =                                     %余料总消耗
  56.6667
≫fx2 = sum(x2)                              %使用钢管总数与解法一、二相同
fx2 =
```

266.6667

上述三种解法的比较，见表 10-11

表 10-11　三种解法结果的比较

目标函数	约束条件	最优解（$x_1 \sim x_8$）	总根数
[1；1；1；1；1；1；1；1]	Aeq = beq	57.14　21.43　21.43　47.62　47.62　47.62　11.90　11.90	266.67
[1；1；1；1；1；1；1；1]	A≥b	55.41　22.30　22.30　48.20　48.20　48.20　11.04　11.04	266.67
[0.1；0.2；0.4；0.2；0；0.4；0.3；0.5]	Aeq = beq	57.98　20.42　21.60　47.30　48.09　46.63　11.46　13.19	266.67

【例 10-9】　今有4种素食品，其维生素每公斤含量，人每天维生素需要量和食品价格，见表 10-12，求在满足人体每天所需维生素的前提下如何使开支最小。

表 10-12　人每天维生素需要量和食品价格

成分＼食品	甲	乙	丙	丁	每天需要量	单位
维生素 A	1000	1500	1750	3250	4000	国际单位（I.U.）
维生素 B	0.6	0.27	0.68	0.3	1.0	mg
维生素 C	17.5	7.5	0.0	30	30	mg
价格/（元/kg）	1.6	1	1.8	3.0	—	—

解：

设甲、乙、丙、丁 4 种食品每天需要量分别为 x_1、x_2、x_3、x_4，目标函数为 Z，求目标函数最小

$$\min Z = 1.6x_1 + x_2 + 1.8x_3 + 3x_4$$

约束条件为

$$1000x_1 + 1500x_2 + 1750x_3 + 3250x_4 \geqslant 4000$$

$$0.6x_1 + 0.27x_2 + 0.68x_3 + 0.3x_4 \geqslant 1.0$$

$$17.5x_1 + 7.5x_2 + 30x_4 \geqslant 30$$

$$x_1、x_2、x_3、x_4 \geqslant 0$$

```
>>f = [1.6;1.0;1.8;3.0];                    % 设置目标函数
>>A = [-1000 -1500 -1750 -3250; -0.6 -0.27 -0.68 -0.3; -17.5 -7.50 -30];
>>format long                               % 设置 15 位数字格式，否则矩阵元素 0.27、0.68
                                              会进位造成计算误差
>>A                                         % 显示不等式约束矩阵，为了改不等式大于形式为
                                              小于形式故不等式二端乘负号
A =
  1.0e+003*
  -1.00000000000000   -1.50000000000000   -1.75000000000000   -3.25000000000000
```

```
  -0.00060000000000   -0.00027000000000   -0.00068000000000   -0.00030000000000
  -0.01750000000000   -0.00750000000000                   0   -0.03000000000000
≫b=[-4000; -1.0; -30];                          %不等式的常数项
≫lb=zeros(4,1);                                 %决策变量的下限
≫[x,fval]=linprog(f,A,b,[],[],lb)               %解线性规划
Optimization terminated successfully.            %最优化成功
x=                                               %最优解
  0.71753681400519
  2.02588130285875
  0.00000000004492
  0.07496653282945
fval=                                            %最优解时的目标值
  3.39883980383625
≫-A*x                                            %核对约束条件满足情况
ans=
  1.0e+003*
  4.00000000006763
  0.00100000000005
  0.03000000000141
```

线性规划计算结果表明，为了达到人所需的维生素，线性规划根据目标函数的要求选取价格低廉的食品。

10.6 整数规划

MATLAB的线性规划解法是基于单纯型法的数值解。因此，它的解题结果往往带有小数。但是，有的实际问题要求决策变量是整数，例如工作人员人数、产品件数、设备台数等。对于这类课题，不能简单地把解题的结果整数化，问题就能解决，这将会带来意想不到的误差。

整数规划的解法有分子定界法、割平面法等。读者可以参考有关线性规划的专著。本书提供的方法是用枚举法来解整数规划问题。所谓枚举法是把所有可能的解都计算出来，从中选取最优的。这种方法虽然是机械的，不机灵的。但是由于计算机运算速度快，存储容量大，对运算所花费时间影响不大。但是当决策变量数量相当大时，这种枚举法显然是不行了。这时只得求助于专用软件，例如商业量化系统软件 QSB 去处理。用枚举法解整数规划的一般步骤如下：

(1) 确定决策变量的数量和变化范围。

(2) 用 for-end 语句作决策变量的整型参数变化的循环，若有多个决策变量则要实现多重循环。

(3) 用 if-end 语句作不等式约束和等式约束条件是否满足的判断。

(4) 符合约束条件的一组决策变量，则进行目标函数计算，并进行存储。否则滑过。

(5) 用函数 max 或 min 语句，搜索目标函数的最大值或最小值及相对应的决策变量。

【例 10-10】 求下列整数规划问题：

目标函数 $\max z = 5x_1 + 8x_2$

s.t.[1]

$$x_1 + x_2 \leqslant 6$$
$$5x_1 + 9x_2 \leqslant 45$$
$$x_1、x_2 \geqslant 0，x_1、x_2 \text{为整数}$$

解：

首先试用非整数解法。

```
≫f=[-5;-8];           %目标函数系数向量，由于求极大，故将目标函数乘负号
≫A=[1 1;5 9]          %不等式约束矩阵
A =
    1   1
    5   9
≫b=[6;45];                                      %不等式约束常数项向量
≫[x,fval]=linprog(f,A,b)                        %线性规划求解
Optimization terminated successfully.
x =                                              %最优解
    2.2500
    3.7500
fval =                                           %目标函数最优值为41.25
  -41.2500
```

采用枚举法求解。在 M 文件编辑器编制解题程序如下，程序名为 examp_1051

```
clear                                            %清内存
k=1;                                             %设初值
for x1=1:6                                       %设决策变量x1的循环，考虑x1+x2
                                                   ≤6，则x1最大不会超过6
    for x2=1:6                                   %设置x2循环
        if x1+x2<=6 & 5*x1+9*x2<=45              %设置约束条件
            z(k)=5*x1+8*x2;                      %满足约束条件下，计算目标函数
            V(k,1)=x1;V(k,2)=x2;                 %记录对应目标函数的决策变量
        end
    end
end
k=k-1;                                           %返回计算目标函数的项数
[zm,mi]=max(z)                                   %搜索最大值zm和对应的下标数mi
x=V(mi,:)                                        %输出对应最大值的决策变量
```

运行程序

```
≫examp_1051                                      %调用上述程序
```

[1] s.t. 为 subject to 的缩写，意思是受约束于。

```
zm =                                    % 解算结果，zm 为目标函数最优值，
                                          mi 为对应 zm 的下标
          39

mi =

      10

x =                                     % 对应目标函数最优值的决策变量，
      3    3                            即 x1 = 3，x2 = 3
```

如果想查看满足约束条件的可行解表矩阵 Z，则可以将目标函数解算向量 z 和矩阵 V 组合起来，在命令窗口输入如下程序：

```
≫Z = horzcat(z', V)          % 向量 z'与 V 的水平连接，得可行解表，第一列为目标
                                函数，第二、三列为决策变量的对应值

Z =

      13    1    1
      21    1    2
      29    1    3
      37    1    4
      18    2    1
      26    2    2
      34    2    3
      23    3    1
      31    3    2
      39    3    3
      28    4    1
      36    4    2
      33    5    1
```

【例 10-11】 求解下列整数规划问题。

目标函数 $$\max z = 4x_1 + 6x_2 + 2x_3$$

s.t.

$$-x_1 + 3x_2 \leqslant 8$$

$$-x_2 + 3x_3 \leqslant 10$$

$$5x_1 - x_3 \leqslant 8$$

$$x_1、x_2、x_3 \geqslant 0，x_1、x_2、x_3 \text{ 为整数}$$

解：

首先试用非整数解法。

```
≫f = [-4; -6; -2];                   % 输入目标函数系数，由于求极大，故乘负号
≫A = [-1 3 0;0 -1 3;5 0 -1]         % 输入不等式约束矩阵
A =

    -1     3     0
     0    -1     3
     5     0    -1
```

```
≫b = [8;10;8];                          %输入常数项向量
≫[x,fval] = linprog(f,A,b)              %规划求解
Optimization terminated successfully.
x =                                     %最优解
    2.5000
    3.5000
    4.5000
    fval =                              %对应最优解的目标函数
  -40.0000
```

采用枚举法求解。在 M 文件编辑器编制解题程序如下：

此程序与 examp_1051 相似，只是增加了决策变量

```
clear
k = 1;
for x1 = 1:8
    for x2 = 1:8
        for x3 = 1:8
            if -x1 + 3* x2<=8 & -x2 + 3* x3<=10 & 5* x - x3<=8
              z(k) = 4* x1 + 6* x2 + 2* x3;
              V(k,1) = x1; V(k,2) = x2; V(k,3) = x3;
              k = k + 1;
            end
        end
    end
end
k = k - 1;
[zm,mi] = max(z)
x = V(mi,:)
```

运行程序

```
≫ examp_1052
zm =                                    %目标函数最大值
    34
mi =
    19
x =                                     %决策变量为 x1 = 2,x2 = 3,x3 = 4
    2   3   4
```

10.7 0-1 规划

0-1 规划是整数规划的一种特殊形式。在这种形式下，决策变量只取 0，1 两个值。例如项目投资，产品的选择，设备的选购，雇员的聘用，投标单位的选取，学生的选取，股票

的选择等，凡是涉及到选取或不选取时，都可以用 0-1 来表示。1 表示选中，而 0 则表示未被选中。初看起来，0-1 规划是比较简单的。但是当决策变量数目相当大时，最优解的搜索也需要花一番功夫，而采用线性规划函数 linprog 已经难以奏效。线规划性则要求逐步缩小搜索区，最后找到最优解。

【例 10-12】 今有一集装箱，拟运输下列物品 A 1 ~ A5，A1、A2 由于体积大，集装箱内只能装其中之一。A4、A5 由于重量大也只能装一件。A1 是食品不能与化工产品 A4 放在一起。A2 与 A5 是配套产品，必需一起运输。A1 的运费是 1500 元，A2 的运费是 2000 元，A3 的运费为 1300 元，A4 的运费是 2300 元，A5 的运费是 2800 元。问集装箱应如何装箱才能使这运费收入达到极大。

解：

设 A1 ~ A5 是否装运的控制变量是 $x_1 \sim x_5$，$x_i = 0$，表示物品 Ai 不装，$x_i = 1$，表示装箱运输。山此目标函数为 Z

$$\max Z = 1.5x_1 + 2x_2 + 1.3x_3 + 2.3x_4 + 2.8x_5$$

s.t.

$x_1 + x_2 \leqslant 1$　　%二者取一

$x_4 + x_5 \leqslant 1$　　%二者取一

$x_2 = x_5$　　%同时装箱或同时不装

$x_1 \neq x_4$　　%互相排斥

x_1、x_2、x_3、x_4、x_5 为 0、1 元素

用枚举法解本题，编制程序如下，程序名为 tranprog。

```
clear                      %清内存
k = 1;Z = 0;               %设初值
for x1 = 0:1               %设置控制变量 x1 循环
    for x2 = 0:1           %设置控制变量 x2 循环
        for x3 = 0:1       %设置控制变量 x3 循环
            for x4 = 0:1   %设置控制变量 x4 循环
                for x5 = 0:1   %设置控制变量 x5 循环
                    if x1 + x3<=1 & x4 + x5<=1 & x4~ = x1 & x2 = = x5
                                                        %设置条件约束
                        Z(k) = 1.5* x1 + 2* x2 + 1.5* x3 + 2.3* x4 + 2.8* x5;
                        Z = [Z;Z(k)];                   %计算目标函数
                        p(k,:) = [x1 x2 x3 x4 x5];k = k + 1;%记录控制变量
                    end
                end
            end
        end
    end
end
[Zmax,i] = max(Z)          %寻找最大值及相应下标
```

```
x = p(i,:)                              %输出控制变量的值运行程序
>>tranprog                              %调用程序
Zmax =                                  %目标函数最大值,6.3 千元
     6.3000
i                                       %符合约束条件的下标
     4
x =                                     %目标函数最大时的控制变量,x1 = x2 = x5 = 1,其余为零
     1    1    0    0    1
```

【例 10-13】 求解0-1规划：已知目标函数

$$\max\ Z = x_1 + 2x_2 + 2x_3 - 6x_4 - 4x_5$$

s.t.

$$3x_1 + 2x_2 - x_3 + 3x_4 + 2x_5 \leqslant 5$$

$$2x_1 + 4x_2 - 2x_3 - x_4 - 2x_5 \leqslant 5$$

$$x_1、x_2、x_3、x_4、x_5 = 0 \text{ 或 } 1$$

解：

(1) 使用一般的数值解法

```
>>f = [-1; -2; -2;6;4];                    %输入目标函数
>>A = [3 2 -1 3 2;2 4 -2 -1 -2]            %输入不等式约束矩阵
>>b = [-5; -5];                            %输入常数向量
>>[x,fval] = linprog(f,A,b)                %线性规划求解
Exiting: One or more of the residuals, duality gap, or total relative error has stalled:
         the dual appears to be infeasible (and the primal unbounded).
         (The primal residual < TolFun = 1.00e-008.)
                                           %退出原因，一个或多个残差，对偶性差距或总
                                             的关系错误而退出。双重显示是不适合的
x =
   1.0e+008*
     3.7774
     0.0017
     4.8601
     2.1585
    -0.0000
fval =
   -2.6452e+009
```

(2) 使用枚举法

在 M 文件编辑器中编制程序如下，程序名为 progzero。

```
clear                                      %清内存
k = 1;Z = 0;                               %设置初值
for x1 = 0:1                               %设置控制变量循环
```

```
            for x2 = 0:1
                for x3 = 0:1
                    for x4 = 0:1
                        for x5 = 0:1
if 3* x1 + 2* x2 - x3 + 3* x4 + 2* x5<=5 & 2* x1 + 4* x2 - 2* x3 - x4 - 2* x5<=5
                                                          %设置条件约束
                    Z(k) = x1 + 2* x2 + 2* x3 - 6* x4 - 4* x5;Z = [Z;Z(k)];
                                                          %计算目标函数
                        p(k,:) = [x1 x2 x3 x4 x5];k = k + 1;
                                                          %存储控制变量
                        end
                    end
                end
            end
        end
end
[Zmax,i] = max(Z)                                         %搜索极大值和下标
x = p(i,:)                                                %显示极大值时的控制变量
```

(3) 运行程序

```
>>progzero
Zmax =                                                    %最大值
     5
i =
     20
x =                                                       %对应最大值的控制变量
                                                          x1 = x2 = x3 = 1,其余为零
     1   1   1   0   0
```

10.8 指派问题

由 0-1 规划派生出来的问题是指派问题。在实际工作中经常会遇到这样的问题，某工程有 n 项任务，需要有 n 个人去完成，每人只能做一件。由于每个人的能力不同，完成每项任务所需时间，费用也不同。因此就产生任务指派问题，指定某人去完成某项任务，使完成任务的效率最高或总的时间最省或总的费用最小。指派问题的数学模型如下：

设第 i 个人去完成第 j 项任务所需费用（或时间）为 C_{ij}，由 C_{ij}元素构成的方阵 C 称为效益矩阵。再设决策变量 X_{ij}，表示第 i 个人是否参与 j 项任务，若参与则 $X_{ij}=1$，否则 $X_{ij}=0$。由 X_{ij}所构成的矩阵 $\boldsymbol{X}$ 称为决策矩阵。则目标函数 Z 为求极小

$$\min Z = \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij}X_{ij}$$

s.t. $\sum_{i=1}^{n} X_{ij} = 1, (j = 1, 2, \cdots, n)$ %表示每人做一项任务

$\sum_{j=1}^{n} X_{ij} = 1, (i = 1, 2, \cdots, n)$ %表示每项任务有人做

$X_{ij} = 0$ 或 1

这个问题是由匈牙利数学家考尼格（Konig）给予解决的，称为匈牙利解法。匈牙利解法的条件为，求目标函数最小，效益矩阵为方阵，效益矩阵中所有元素大于等于零。匈牙利解法的重要定理的叙述如下，这里不加以证明，有兴趣的读者可查阅线性规划的专著。

【定理 10-1】 从效益矩阵 C 的第 k 行（或 k 列）的每一个元素中减去一个常数得到新的效益矩阵 $\boldsymbol{C}_1$ 所表示的指派问题与原效益矩阵 $\boldsymbol{C}$ 具有相同的最优解。

【例 10-14】 今有 W、X、Y、Z 4种工件由工人 A、B、C、D 去加工，加工时间见表 10-13 效益矩阵，试简化效益矩阵，使尽可能增加零元素。并求最优指派任务使总的加工时间最小。

表 10-13 某个零件由 4 个工人加工的效益矩阵

人 \ 工件	W	X	Y	Z
A	9	15	7	5
B	13	11	18	7
C	6	8	13	17
D	11	4	19	10

解：

山于效益矩阵元素不能为负，故用每行（或每列）的最小元素去减每行（或每列）的其他元素。使效益矩阵得到更多的零元素。

在 M 文件编辑器中编制简化效益矩阵的程序如下：

```
function Q = effmat(A)                %函数定义行，A 为效益矩阵，函数名为 effmat
n = size(A,1);                        %提取效益矩阵的行数
for i = 1:n                           %设置行循环
    Q(i,:) = A(i,:) - min(A(i,:))* ones(1,n); %在每行中，减去行中最小元素
end
for j = 1:n                           %设置列循环
    Q(:,j) = Q(:,j) - min(Q(:,j))* ones(n,1); %在每列中，减去列中最小元素
End
```

运行程序

```
>>A = [9 15 7 5;13 11 18 7;6 8 13 17;11 4 19 20]%输入效益矩阵
A =
    9   15    7    5
   13   11   18    7
    6    8   13   17
   11    4   19   20
```

```
≫effmat(A)                        %简化效益矩阵
ans =
     4    10     0     0
     6     4     9     0
     0     2     5    11
     7     0    13    16
```

简化效益矩阵后，每行每列都有了零元素，如何取得最优解，还得利用下面的定理 2。

【定理 10-2】 若方阵中部分元素为零，部分元素为非零，则覆盖方阵内所有零元素的最小直线数，等于矩阵中独立零元素的最多个数，则最优解成立。

所谓独立零元素是处于不同行，不同列上的零元素。如果得到了独立零元素，则将独立零元素所对应的位置取 1，将其他元素取零，由此得到的分配问题就是最优解。在上例中经化简了的效益矩阵，已经得到独立零元素。将上述 ans 矩阵进行如下操作即得到最优决策变量。

```
≫X(1,3) = 1;X(2,4) = 1;X(3,1) = 1;X(4,2) = 1;%将独立零元素设为 1
≫X
X =
     4    10     1     0
     6     4     9     1
     1     2     5    11
     7     1    13    16
≫for i = 1:4                                   %将非 1 元素设为零
     for j = 1:4
         if X(i,j)~ = 1
            X(i,j) = 0;
         end
       end
  end,X
X =                                            %得决策变量的最优解,即取 x13 =
                                                x24 = x31 = x42 = 1
     0     0     1     0
     0     0     0     1
     1     0     0     0
     0     1     0     0
≫A = [9 15 7 5;13 11 18 7;6 8 13 17;11 4 19 20]%重新输入效益矩阵
A =
      9    15     7     5
     13    11    18     7
      6     8    13    17
     11     4    19    20
```

```
>>minZ = sum(sum(A.* X))                    %计算目标函数最小值,最优解完成
minZ =
    24
```

本例中采用化简效益矩阵后，就出现独立零元素的情况，是比较少见的。往往需要多次调整。一般解题需要经过4步。

(1) 将行（或列）元素用同行（或列）最小元素相减。

将每行（或每列）的最小元素去减每行（或每列）的其他元素。使效益矩阵含有较多的零元素。

(2) 进行最优性检验。

用最少的直线去覆盖效益矩阵中的零元素。如果覆盖的直线数恰好等于效益矩阵的阶数，则此矩阵即为最优解。否则应进行调整。

(3) 调整方法。

用最少的直线，划去效益矩阵中零元素，在留下的矩阵中寻找最小元素，用此元素与未划去的元素相减，以增加零元素。然后在水平所划直线与垂直所划直线的交叉点上，加上这个最小元素。调整后，再进行最优性检验，直到每行、每列存在有独立的零元素为止。

(4) 进行指派，以确定最优解。

1) 先从只有一个零元素的行开始。给零元素加标记，并将其所在列的其他零元素划去。

2) 再从只有一个零元素的列开始，给零元素加标记，并将其所在行的其他零元素划去。

3) 反复进行以上两步，直到所有的零元素都被加上标记或被划去。

4) 当标记了的零元素是独立的零元素时，则找到了最优解。

【例10-15】 今有4台机床M1～M4，分别加工4种零件P1～P4，其加工所需时间见表10-14，若每台机床只限加工一种另件，问如何指派任务才能使总的加工工时最小。

表10-14　4台机床分别加工4种零件所需时间

机床＼零件	P1	P2	P3	P4
M1	2	10	9	7
M2	15	4	14	8
M3	13	14	16	11
M4	4	15	13	9

解：

```
>>A = [2 10 9 7;15 4 14 8;13 14 16 11;4 15 13 9]
A =                                         %输入效益矩阵
     2    10     9     7
    15     4    14     8
    13    14    16    11
     4    15    13     9
>>effmat(A)                                 %等值化简
ans =
     0     8     2     5
    11     0     5     4
     2     3     0     0
     0    11     4     5
```

```
≫Q = ans;                                              %存储
≫Ad23 = Q(2:3,:)                                      %准备删去含有零元素的两行
Ad23 =
     11   0   5   4
      2   3   0   0
≫Bd1 = Q(:,1)                                         %准备删去含有零元素的列
Bd1 =
     0
     11
     2
     0
≫Q(2:3,:) = []; Q(:,1) = [];                          %最优检验,划零线数<4,非最优
≫Q
Q =                                                    %剩余矩阵
      8   2   5
     11   4   5
≫q = min(Q(:))                                        %提取最小值
q =
     2
≫Q = Q - q* ones(size(Q))                             %剩余矩阵减 q
Q =
     6   0   3
     9   2   3
≫Bd1(2:3) = Bd1(2:3) + q* ones(size(Bd1(2:3)))       %交义元素加 q
Bd1 =
     0
     13
     4
     0
≫B2 = [Q(1,:); Ad23(:,2:4); Q(2,:)]                  %重组矩阵
B2 =
     6   0   3
     0   5   4
     3   0   0
     9   2   3
≫B = [Bd1,B2]                                         %最优性检验成功,每行、每列有独立
                                                       的零元素
B =
      0   6   0   3
     13   0   5   4
```

```
    4   3   0   0
    0   9   2   3
>>B(2,2) = 1;B(3,4) = 1;B(1,3) = 1;B(4,1) = 1; %独立零元数置 1
>>X = B                                          %矩阵 B 赋入矩阵 X
X =
     0   6   1   3
    13   1   5   4
     4   3   0   1
     1   9   2   3
>>for i = 1:4                                    %实现对矩阵 X 的非 1 元素置零
    for j = 1:4
        if X(i,j)~ = 1
          X(i,j) = 0;
        end
    end
  end,X
X =                                              %决策变量 X13 = X22 = X34 = X41 = 1
    0   0   1   0
    0   1   0   0
    0   0   0   1
    1   0   0   0
>>A = [2 10 9 7;15 4 14 8;13 14 16 11;4 15 13 9]%重新输入效益矩阵
A =
     2    10    9     7
    15     4   14     8
    13    14   16    11
     4    15   13     9
>>minZ = sum(sum(A.* X))                         % 效益矩阵与决策矩阵的数组乘积，
                                                   再取列元素之和，再取行元素之和
                                                   即为目标函数最优值
minZ =
    28
```

10.9 指派问题的猜想

作者发现，当效益矩阵为魔方矩阵时，只需经一次化简就得到含有独立零元素的矩阵。从 3 阶到 20 阶全部成立。因此想推论对 n 阶也成立。并且计算出最优解

$$\min Z = F(n)$$

与 n 的函数关系。下面只列举 n = 5、6、7 的情况。

```
>>n = 5;
>>A = magic(n)                    %输入效益矩阵
```

```
A =
    17   24    1    8   15
    23    5    7   14   16
     4    6   13   20   22
    10   12   19   21    3
    11   18   25    2    9
≫B = effmat(A)                          %进行简化,已经生成独立零元素
B =
    16   23    0    7   14
    18    0    2    9   11
     0    2    9   16   18
     7    9   16   18    0
     9   16   23    0    7
≫B(1,3) = 1;B(2,2) = 1;B(3,1) = 1;B(4,5) = 1;B(5,4) = 1;
                                        %对独立零元素置 1
≫magmin(A,B)                            %计算决策变量及最优值,magmin 程序附后
X =                                     %决策矩阵
    0   0   1   0   0
    0   1   0   0   0
    1   0   0   0   0
    0   0   0   0   1
    0   0   0   1   0
minZ =                                  %最优值
    15
≫n = 6;
≫A = magic(n)                           %6 阶效益矩阵
A =
    35    1    6   26   19   24
     3   32    7   21   23   25
    31    9    2   22   27   20
     8   28   33   17   10   15
    30    5   34   12   14   16
     4   36   29   13   18   11
≫B = effmat(A)                          %简化
B =
    34    0    5   18   16   16
     0   29    4   11   18   15
    29    7    0   13   23   11
     0   20   25    2    0    0
```

```
      25    0   29   0    7    4
       0   32   25   2   12    0
≫B(1,2) = 1;B(3,3) = 1;B(5,4) = 1;B(4,5) = 1;B(2,1) = 1;B(6,6) = 1;
                                        %设置独立变量
≫magmin(A,B)                            %计算决策变量与最优值
X =                                     %决策变量
     0   1   0   0   0   0
     1   0   0   0   0   0
     0   0   1   0   0   0
     0   0   0   0   1   0
     0   0   0   1   0   0
     0   0   0   0   0   1
minZ =                                  %最优解
     39
当 n = 7
≫magmin(A,B)
X =                                     %决策变量
     0   0   0   1   0   0   0
     0   0   1   0   0   0   0
     0   1   0   0   0   0   0
     1   0   0   0   0   0   0
     0   0   0   0   0   0   1
     0   0   0   0   0   1   0
     0   0   0   0   1   0   0
minZ =                                  %最优解
     28
```

计算决策变量矩阵和目标函数最优解的程序如下：

```
function z = magmin(A,B)                % 函数定义项，它由效益矩阵 A 和独立零元素矩
                                          阵 B 确定
n = size(B,1);                          %测试矩阵 B 的阶数
X = B;                                  %赋予决策矩阵 X
for i = 1:n                             %设置循环
    for j = 1:n
        if X(i,j) ~ = 1                 %是 1 元素保留，非 1 则置零
            X(i,j) = 0;
        end
    end
end
X                                       %输出决策矩阵
```

```
minZ = sum(sum(A.* X))                 %输出目标函数最优值
```

10.10 指派问题的枚举法

上述指派问题也可能用枚举法来解。若效益矩阵的阶数不大，则可以通过枚举法去搜索最优值。枚举法对每一种指派方案计算它的价值。随后用 max 或 min 搜索它的最优值。作者编制了 4 ~ 7 阶的枚举法指派程序如下，程序名为 zolp。

```
% This prog for solution the problem of allot
A = input('Input matrix of the allot problem \ n')  %输入效益矩阵
ud = input('Input 0 for min,input 1 for max \ n')  %输入求极大还是求极小
n1 = size(A,1);n11 = size(A,2);
if n1 ~ = n11                                       %若行数与列数不等,则终止程序
    disp('matrix must be square'),break
end
switch n1                                           %分别不同的阶次
    case 4
        arrange_4                                   %分配 4 阶组合的子程序
    case 5
        arrange_5                                   %分配 5 阶组合的子程序
    case 6
        arrange_6                                   %分配 6 阶组合的子程序
    case 7
        arrange_7                                   %分配 7 阶组合的子程序
    otherwise                                       %否则中断
        disp('the matrix order must be less then seven'),break
end
        n2 = prod(1:n1)                             %计算搜索次数
S = zeros(1,n2);                                    %设置目标函数的列元数
for t = 1:n2
    switch n1
      case 4                                        %对于 4 阶指派问题
          b1 = C(t,2);b2 = C(t,3);b3 = C(t,4);b4 = C(t,5);
          S(t) = sum(A(1,b1) + A(2,b2) + A(3,b3) + A(4,b4));
      case 5                                        %对于 5 阶指派问题
          b1 = C(t,2);b2 = C(t,3);b3 = C(t,4);b4 = C(t,5);b5 = C(t,6);
          S(t) = sum(A(1,b1) + A(2,b2) + A(3,b3) + A(4,b4)) + A(5,b5);
      case 6                                        %对于 6 阶指派问题
          b1 = C(t,2);b2 = C(t,3);b3 = C(t,4);b4 = C(t,5);b5 = C(t,6);b6 = C(t,7);
          S(t) = sum(A(1,b1) + A(2,b2) + A(3,b3) + A(4,b4)) + A(5,b5) + A(6,
                b6);
```

```
        case 7                                          %对于7阶指派问题
b1 = C(t,2); b2 = C(t,3); b3 = C(t,4); b4 = C(t,5); b5 = C(t,6); b6 = C(t,7); b7 = C(t,
    8);
S(t) = sum(A(1,b1) + A(2,b2) + A(3,b3) + A(4,b4)) + A(5,b5) + A(6,b6) + A(7,b7);
            end
    end
    if ud = = 1                                          %求目标函数最大时的函数值及下标
    [pm,t] = max(S)
    elseif ud = = 0                                      %求目标函数最小时的函数值及下标
    [pm,t] = min(S)
else                                                     %否则输入错误
  disp('input error')
end
x = C(t,:)                                               %输出最优决策变量
```

下面是4、5、6、7阶的子程序

arrange_4的程序

```
% This program is set a four order arrange
t = 1; C = zeros(1,5);
for i = 1:4
    for j = 1:4
        for k = 1:4
            if k~ = i & i~ = j & j~ = k
              V = [1,2,3,4]; V(i) = 0; V(j) = 0; V(k) = 0; p = find(V);
              B = [t,i,j,k,p]; t = t + 1; C = [C;B];
            end
        end
    end
end
C(1,:) = [];
```

Arrange_5的程序

```
% This program is set a five order arrange
t = 1; C = zeros(1,6);
for i = 1:5
    for j = 1:5
        for k = 1:5
            for l = 1:5
                if l~ = i & l~ = j & l~ = k & i~ = j & i~ = k & j~ = k
                V = [1,2,3,4,5]; V(i) = 0; V(j) = 0; V(k) = 0; V(l) = 0; p = find(V);
                B = [t,i,j,k,l,p]; t = t + 1;
```

```
                C = [C;B];
                end
            end
        end
    end
end
C(1,:) = [];
```

Arrange_6 的程序

```
% This program is set a six order arrange
t = 1;C = zeros(1,7);
for i = 1:6
    for j = 1:6
        for k = 1:6
            for l = 1:6
                for m = 1:6
        if m~ = i & m~ = j & m~ = k & m~ = l & i~ = j & i~ = k & i~ = l & j~ = k &
          j~ = l & k~ = l
V = [1,2,3,4,5,6];V(i) = 0;V(j) = 0;V(k) = 0;V(l) = 0;V(m) = 0;p = find(V);
                    B = [t,i,j,k,l,m,p];t = t + 1;
                    C = [C;B];
                    end
                end
            end
        end
    end
end
C(1,:) = [];
```

Arrange_7 的程序

```
% This program is set a seven order arrange
t = 1;C = zeros(1,8);
for i = 1:7
    for j = 1:7
        for k = 1:7
            for l = 1:7
                for m = 1:7
                    for n = 1:7
  if n~ = i & n~ = j & n~ = k & n~ = l & n~ = m & i~ = j & i~ = k & i~ = l & i~ = m
    & j~ = k & j~ = l & j~ = m & k~ = l & k~ = m & l~ = m
V = [1,2,3,4,5,6,7];V(i) = 0;V(j) = 0;V(k) = 0;V(l) = 0;V(m) = 0;V(n) = 0;p = find
```

```
(V);
                    B = [t,i,j,k,l,m,n,p];t = t + 1;
                    C = [C;B];
                    end
                  end
                end
              end
            end
          end
end
C(1,:) = [];
```

【例 10-16】 某翻译公司有5位高级翻译，简称 A、B、C、D、E，每位掌握 5 国文字，即英、俄、德、日、法。经测试他们的翻译速度（字数/小时）有差异，见表 10-15。问应如何指派他们成专职翻译使总的翻译效率最高。若 D 对翻译德语有困难，E 对翻译法语有困难时应如何调整。

表 10-15　某公司 5 位高级翻译的翻译速度

语种 翻译	英	俄	日	德	法
A	950*	760	610	400	550
B	850	450	790	930*	350
C	600	750	920*	910	810
D	480	820	730	410	940*
E	570	920*	680	890	510

解：

```
≫zolp
Input matrix of the allot problem                %输入指派问题的矩阵
[950 760 610 400 550;850 450 790 930 350;600 750 920 910 810;480 820 730 410 940;
 570 920 680 890 510]
A =
950   760   610   400   550
850   450   790   930   350
600   750   920   910   810
    480   820   730   410   940
    570   920   680   890   510
Input 0 for min,input 1 for max                  %输入 1,求极大
1
ud =
    1
n2 =                                             %计算 120 种方案
```

```
    120
pm =                                         %5 人每小时翻译字数
    4660
t =                                          %最优值向量的下标
    16
x =                                          %最优决策变量。翻译 A 从事英文,翻译 B
                                               从事德文,翻译 C 从事日文,翻译 D 从事
                                               法文,翻译 E 从事俄文。表 10-15 中标有
                                               * 者表示选中
    16    1    4    3    5    2
```

x 向量的定义如下:

x (1) ——目标函数的下标数;

x (2) ——决策矩阵的第 1 行列数;

x (3) ——决策矩阵的第 2 行列数;

x (4) ——决策矩阵的第 3 行列数;

x (5) ——决策矩阵的第 4 行列数;

x (6) ——决策矩阵的第 5 行列数。

对于 D 对翻译德语有困难,E 对翻译法语有困难时,无须进行调整,对最优化指派无影响。

【例 10-17】 某人事部门拟从干部 A、B、C、D、E、F 中选拔到 P1、P2、P3、P4 部门的领导岗位,经群众评议,6 名干部得分见表 10-16,问应如何选取使总的得分最高。

表 10-16 某人事部门 6 名干部得分

部门 \ 人员	A	B	C	D	E	F
P1	0.95*	0.76	0.61	0.95	0.56	0.48
P2	0.23	0.45	0.79	0.93*	0.35	0.44
P3	0.6	0.47	0.92*	0.92	0.81	0.6
P4	0.48	0.91	0.73	0.41	0.9*	0.68
P5	0.89	0.89*	0.44	0.89	0.56	0.37

解:

由于指派问题的枚举法程序设计是针对效益矩阵为方阵,而本题是行列不对称的非标准形式。为此可以虚设一行全为零元素的 P6 来补缺。以下是程序运行结果。

```
≫zolp                                        % 调用指派问题的枚举法程序
Input matrix of the allot problem
A                                            %输入矩阵 A,矩阵 A 在程序调用前已定义
A =                                          %程序响应
```

```
    0.9500  0.7600  0.6100  0.9500  0.5600  0.4800
    0.2300  0.4500  0.7900  0.9300  0.3500  0.4400
    0.6000  0.4700  0.9200  0.9200  0.8100  0.6000
    0.4800  0.9100  0.7300  0.4100  0.9000  0.6800
    0.8900  0.8900  0.4400  0.8900  0.5600  0.3700
         0       0       0       0       0       0
Input 0 for min, input 1 for max            %询问求极大还是求极小
1                                           %输入 1，求极大
ud =                                        %程序响应
     1
n2 =                                        %搜索次数
     720
pm =                                        %目标函数最大值
     4.5900
t =                                         %最大值出现时，目标函数向量的下标
     57
x =                                         %决策向量，对应决策矩阵，表 10-16 中标
                                              有 * 者表示选中的组合。这 5 个数之和恰
                                              好等于目标函数的最大值
     57   1   4   3   5   2   6
```

由此指派部门 P1 的领导由 A 担任；部门 P2 的领导由 D 担任；部门 P3 的领导由 C 担任；部门 P4 的领导由 E 担任；部门 P5 的领导由 B 担任；干部 F 落选。

对于类似的非标准指派问题，也同样可以用虚设的行或列来补充，而虚设的行或列的元素应为零，它对目标函数的值无影响。

读者可以在运行 zolp 程序后，查看目标函数向量 S，对于 6 阶指派问题，它扫描了 720 项，由 max 函数找出它的最大值，再从这最大值中找出它的指派组合。对本例在命令窗口输入如下程序：

```
≫S(57)                                      %显示目标函数的最大值
ans =
     4.5900
≫C(57,:)                                    %显示目标函数为最大值时的组合
ans =
     57   1   4   3   5   2   6
```

第 10 章习题

10-1　求解线性规划问题

$$\max Z = 2x_1 + 7x_2 + 3x_3$$

s.t.

$$x_1 + 3x_2 + 4x_3 \leq 30$$

$$x_1 + 4x_2 + x_3 \leq 20$$

$$x_1、x_2、x_3 \geqslant 0$$

10-2　求解具有等式压束的线性规划问题

$$\max z = 5x_1 + 3x_2 + 2x_3 + x_4$$

s.t.

$$5x_1 + x_2 + 2x_3 + 7x_4 = 12$$

$$2x_1 + 4x_2 + 3x_3 + x_4 = 10$$

$$x_1、x_2、x_3、x_4 \geqslant 0$$

10-3　求解线性规划问题

$$\min\ z = 2x_1 + 3x_2 + x_3$$

s.t.

$$x_1 + 4x_2 + 2x_3 \geqslant 24$$

$$3x_1 + 2x_2 \geqslant 18$$

$$x_1、x_2、x_3 \geqslant 0$$

10-4　求解具有等式压束和不等式约束的线性规划问题

$$\min\ z = 3x_1 - 5x_2 + 2x_3 - x_4$$

s.t.

$$2x_1 + x_2 + 2x_3 + x_4 = 20$$

$$x_3 + 2x_4 \leqslant 20$$

$$x_1 - x_2 + x_3 \geqslant 1$$

$$6 < 2x_1 + 3x_2 + x_3 + x_4 < 28$$

$$x_1,\ x_2,\ x_3,\ x_4 \geqslant 0$$

10-5　求解线性规划问题

$$\min\ z = 30x_1 + 20x_2$$

s.t.

$$x_1 + x_2 \geqslant 2$$

$$3x_1 + 4x_2 \geqslant 7$$

$$4x_1 + x_2 \geqslant 3$$

$$x_1,\ x_2 \geqslant 0$$

10-6　求解线性规划问题，有一个变量无约束。

$$\min\ z = 8x_1 + 3x_2 + 6x_3$$

s.t.

$$x_1 + 2x_2 + x_3 \leqslant 18$$

$$2x_1 + x_2 + 3x_3 \leqslant 16$$

$$x_1 + x_2 + x_3 = 10$$

$$x_1、x_2 \geqslant 0,\ x_3 \text{ 无约束}$$

提示：下限设置 lb = [0; 0; -inf]

10-7　求解线性规划问题

$$\max z = x_1 + x_2 + x_3 + x_4 + x_5 + x_6$$

s.t. $x_1 + 2x_2 \leqslant 9$； $x_2 + 2x_3 \leqslant 11$； $x_3 + 2x_4 \leqslant 13$； $x_4 + 2x_5 \leqslant 15$； $x_5 + 2x_6 \leqslant 17$； x_1， x_2， …， $x_6 \geqslant 0$

10-8　求解线性规划问题

$$\min z = x_1 + x_2 + x_3$$

s.t.

$$x_1 + x_4 + 2x_5 - 3x_6 = 4$$

$$x_2 + x_4 + 2x_5 - 4x_6 = 7$$

$$x_3 + x_4 + 2x_5 - 5x_6 = 11$$

$$x_1、x_2、x_3、x_4、x_5、x_6 \geqslant 0$$

提示：设置目标函数系数 f = [1;1;1;0;0;0]，下限设置 lb = [0; 0; 0; 0; 0; 0];

10-9 某箱体制造厂，需要生产 A，B，C 三种类型箱体，每种箱体单台所需角钢的根数和长度见表 10-17。已知钢材的来料长度为 3.5 米。今拟生产 A、B 型的箱体各 100 台，C 型箱体 50 台。问应如何下料，使角钢所用的根数最少，共需多少根。

表 10-17 每种箱体单台所需角钢的根数和长度

箱体 \ 长度/m	0.4	0.6	0.7	1.0	1.2	1.8	2.3
A	8	8			4		
B			8	8		4	
C			8	8			4

10-10 求解整数规划

$$\max z = 5x_1 + 7x_2$$

s.t.

$$x_1 + x_2 \leqslant 10$$

$$3x_1 + 5x_2 \leqslant 20$$

$$x_1、x_2 \text{ 为正整数}$$

10-11 求解 0-1 规划

$$\min z = 4x_1 + 3x_2 + 2x_3$$

s.t.

$$2x_1 - 5x_2 + 3x_3 \leqslant 5$$

$$4x_1 + x_2 + 3x_3 \geqslant 3$$

$$x_1、x_2、x_3 = 0 \text{ 或 } 1$$

10-12 已知效益矩阵 $\boldsymbol{A}$

$$\boldsymbol{A} = \begin{bmatrix} 95 & 76 & 61 & 40 & 54 \\ 35 & 45 & 79 & 93 & 35 \\ 60 & 71 & 92 & 91 & 81 \\ 48 & 82 & 73 & 41 & 69 \\ 89 & 44 & 47 & 89 & 36 \end{bmatrix}$$

求指派后的总分最大。

10-13 某厂有 6 台机床 m1 ~ m6，分别加工 6 种零件所需加工时间见表 10-18。今拟每台机床专职加工一种零件，问应如何分配使总的加工时间最短。

表 10-18 某厂 6 台机床分别加工 6 种零件所需加工时间

零件 \ 机床	m1	m2	m3	m4	m5	m6
P1	20	74	52	37	18	69
P2	55	44	20	83	19	37
P3	60	39	67	50	68	86
P4	27	46	83	70	30	85
P5	34	41	36	42	54	59
P6	48	84	68	30	15	49

第 11 章　MATLAB 在自动控制中的应用

自动控制是应用数学工具比较多的技术领域。自动控制中所应用的数学包括矩阵、微分方程稳定性理论、复变函数理论、最优化理论和模糊数学等。在 MATLAB 未产生前，由于分析自动控制系统的复杂性，计算工作量大，所以在过去，分析一个系统的稳定性和参数优化，往往要花费相当长的时间，尤其对于多输入多输出系统和非线性控制系统。

采用 MATLAB 以后，稳定性分析和参数优化，将是轻而易举的事，还可以通过 Simulink 来仿真，并选择系统结构和参数，避免可能发生的计算错误。因此大大提高了工作效率。

本章介绍 MATLAB 在自动控制方面的入门，即单输入单输出模型，内容包括传递函数列写，传递函数的串、并联和反馈连接，系统的稳定性分析、根轨迹图、博德图、奈奎斯特图、系统的状态空间表示法和比例积分控制的计算等。在自动控制系统的稳定性分析这一节里，作者使用了 MATLAB 中关于计算行列式的函数 det，编制了计算赫尔维茨行列式的程序。这给分析系统的稳定性带来极大的方便。读者不必再使用劳斯判据或逐个输入赫尔维茨行列式元素，只要输入特征方程式系数就可以分析稳定性。

用于自动控制的常用命令和函数见表 11-1。

表 11-1　自动控制系统常用命令和函数

函　数	说　明	所属工具箱
tf	建立传递函数，或转换 LTI 模型成传递函数形式	控制系统
tfdata	快速访问传递函数数据	控制系统
tzero	提取系统模型的零点	控制系统
pole	提取系统模型的极点	控制系统
conv	计算两个多项式的乘积	MATLAB
pzmap	由传递函数或状态空间模型所产生零点、极点的分布图或零、极点的数据	控制系统
zpk	由传递函数或状态空间模型所产生零点，极点和增益形式的传递函数	控制系统
zpkdata	给定一个连续的 LTI 系统，产生模型的零点、极点、增益及相关信息	控制系统
step	给定一个连续的 LTI 系统，由输入单位阶跃控制作用所产生的输出响应	控制系统
impulse	给定一个连续的 LTI 系统，由输入单位脉冲所产生的输出响应	控制系统
dcgain	计算静态增益	控制系统
find	寻找矩阵的非零元素及其下标	MATLAB
lsim	计算任意输入作用下的输出响应	控制系统
residue	计算分式多项式的留数	MATLAB
roots	解代数多项式的根	MATLAB
series	模型的串联连接	控制系统
parallel	模型的并联连接	控制系统
feedback	模型的反馈连接	控制系统
ss	建立状态空间模型	控制系统
rlocus	绘制根轨迹图	控制系统

（续）

函　　数	说　　明	所属工具箱
rlocsfind	为根轨迹图上设置极点，寻找相应的增益	控制系统
canon	将状态空间模型化为标准形式	控制系统
bode	博德图	控制系统
nichols	尼柯尔斯图	控制系统
nyquist	奈奎斯特图	控制系统
gensig	信号产生器，产生正弦波、方波、脉冲波、锯齿波供选择	控制系统
ctrb	给定一个系统的矩阵 **A**、**B**，它返回一个可控矩阵，用来检测系统的可控性	控制系统
place	给定一个系统的状态矩阵 **A**、**B** 和极点配置向量 **P**，它返回增益矩阵 **F**，将 **A** - **B** **F** 的特征值配置在向量 **P** 上	控制系统
acker	与 place 相似	控制系统
obsv	给定一个系统的状态矩阵 **A**、**C**，它返回一个能观测矩阵，用来检测系统的可观测性	控制系统

11.1　传递函数的列写

控制系统设计的第一步是建立系统的模型。有多种方法可以建立系统模型，其中以传递函法和状态空间法用得最为普遍。考虑一个单输入 $u(t)$、单输出 $y(t)$ 系统，它是由下列微分方程式联系着的。

$$y'' + 5y' + 6y = u' + u$$

考虑初始条件为零，对上式进行拉普拉斯变换，由此可得传递函数为输出 $y(s)$ 与输入 $u(s)$之比。

$$G(s) = (s+1)/(s^2+5s+6)$$

如果知道传递函数 $G(s)$ 的分子和分母多项式，就可以在 MATLAB 中将系统模型表示成线性时不变（LTI）的对象，并建立传递函数。传递函数的书写格式为

sys = tf（num，den）

sys1 = tf（num，den，Ts）

式中，sys 为 LTI 对象的传递函数；tf 为创建传递函数命令；num 为分子多项式系数（依降幂排列）；den 为分母多项式系数；sys1 为建立采样系统传递函数，它是以 z 变换形式出现；Ts 为采样时间。

在上述例子中，在 MATLAB 命令窗口中输入如下程序，传递函数即创建完成。

```
≫num = [1 1];                    %分子多项式系数
≫den = [1 5 6];                  %分母多项式系数
≫G = tf(num,den)
Transfer function:                %创建的传递函数
    s + 1
-------------
s^2 + 5 s + 6
≫G1 = tf([1 1],[1 5 6]           %可以直接输入分子、分母多项式系数
```

```
Transfer function:
      s + 1
------------------------
s^2 + 5 s + 6
```

【例 11-1】 考虑一个 4 阶的 LTI 系统，其微分方程式为

$$y^{(4)} + 2.1y^{(3)} + 2.75y^{(2)} + 3.15y^{(1)} + y = 2u^{(2)} + 3u^{(1)} + u$$

式中，初始条件全为零，求其传递函数零极点分布图。

解：对上式进行拉普拉斯变换后得传递函数为

$$G(s) = \frac{2s^2 + 3s + 1}{s^4 + 2.1s^3 + 2.75s^2 + 3.15s + 1}$$，具体程序如下：

```
>>num = [2 3 1];                      %输入 G(s)的分子多项式系数
>>den = [1 2.1 2.75 3.15 1];          %输入 G(s)的分母多项式系数
>>G = tf(num,den)                     %创建传递函数
Transfer function:
              2 s^2 + 3 s + 1
------------------------------------------
s^4 + 2.1 s^3 + 2.75 s^2 + 3.15 s + 1
>>get(G)                              %提取传递函数的信息
              num: {[0 0 2 3 1]}
              den: {[1 2.1 2.75 3.15 1]}
         Variable: 's'
               Ts: 0
          ioDelay: 0
       InputDelay: 0
      OutputDelay: 0
        InputName: {''}
       OutputName: {''}
       InputGroup: {0x2 cell}
      OutputGroup: {0x2 cell}
            Notes: {}
         UserData: []
>>pzmap(G)                            %绘制零、极点分布图，如图 11-1 所示
>>grid on                             %添加坐标栅格线
>>G1 = zpk(G)                         %将 G(s)转换成零点、极点、增益型传递函数
zero/pole/gain:
                2 (s + 1) (s + 0.5)
--------------------------------------------------------------------
(s + 1.381) (s + 0.4431) (s^2 + 0.2762s + 1.634)
>>[z,p,k] = zpkdata(G,'v')            %将 ZPK 模型中的零点、极点、增益列成数组
```

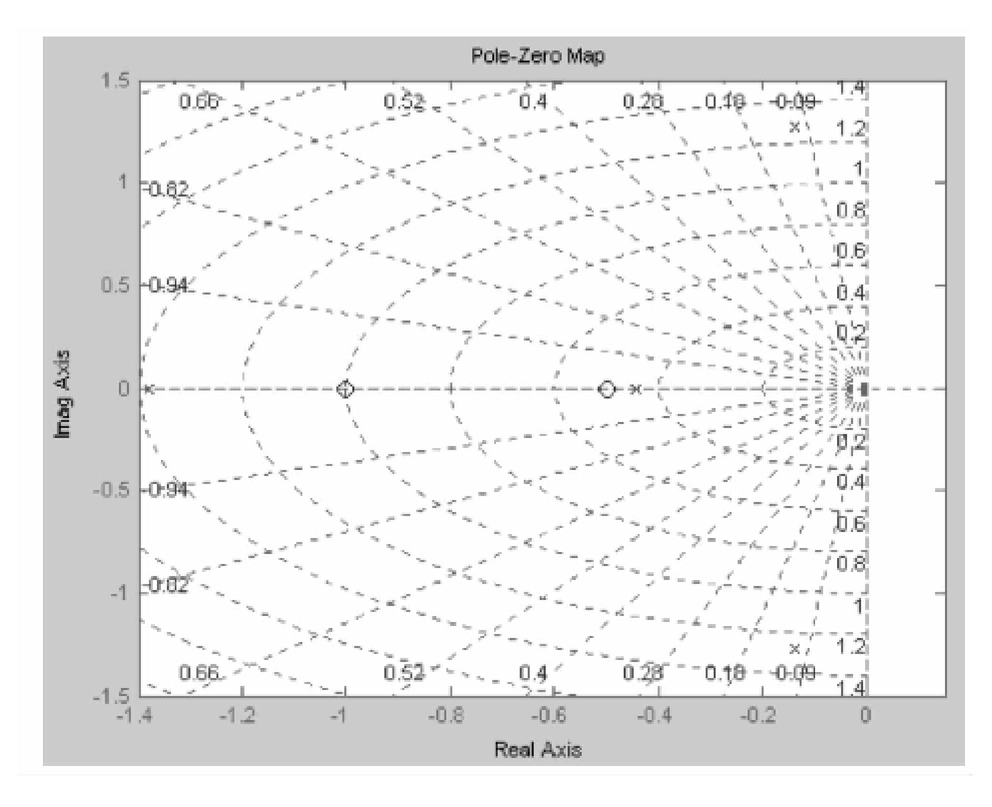


图 11-1　传递函数 G（s）的零极点分布图（圆点为零点，叉号为极点）

```
z =                                        %零点
   -1.0000
   -0.5000
p =                                        %极点
   -0.1381 + 1.2710i
   -0.1381 - 1.2710i
   -1.3807
   -0.4431
k =
     2
```

所以，通过上述简单操作，就得到传递函数的大量信息。首先系统是稳定的，因为极点全在左半平面。系统有一对极点，所以系统带有衰减振荡。由于有一个零点与极点相近，所以对控制作用近似可看成 3 阶系统。

11.2　控制系统的状态表示法

对于单输入单输出系统，模型阶数不超过 4 阶，用传递函数来表示一个系统是合适的。对于高阶系统或者多变量系统，传递函数表示方式则显得困难。因为实际系统通常是用状态变量从物理定律中推导出来，状态变量是与可识别的量相对应的。状态变量可以很方便分析非零初始条件下的系统响应。考虑 SISO（单输入单输出）系统是由 n 阶线性微分方程式来描述。

$$y^{(n)} + a_1 y^{(n-1)} + \cdots + a_{n-1} y' + a_n y = b_0 u^{(n)} + b_1 u^{(n-1)} + \cdots + b_{n-1} u' + b_n u$$

那么，它可以用 n 个一阶线性微分方程组来表示。列写成矩阵形式即为

$$x' = Ax + Bu$$
$$y = Cx + Du$$

式中，x 为状态变量；u 为控制作用向量；A 为状态矩阵

$$A = \begin{vmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \cdots & 1 \\ -a_n & -a_{n-1} & -a_{n-2} & \cdots & -a_1 \end{vmatrix}$$

B 为控制作用系数向量，$B(1) = b_1 - a_1 B(0)$

$$B(2) = b_2 - a_1 B(1) - a_2 B(0)$$
$$B(3) = b_3 - a_1 B(2) - a_2 B(1) - a_3 B(0)$$
$$B(n) = b_n - a_1 B(n-1) - a_2 B(n-2) - \cdots - a_{n-1} B(1) - a_n B(0)$$

C 为输出矩阵向量，对于 SISO 系统 $C = [1\ 0\ \cdots\ 0]$

D 为控制作用的静态增益，$D = b_0 = B(0)$

有了上述状态方程式，就可以在系统的传递函数与状态方程间进行转换，借助于 MATLAB 的转换函数 ss。函数 ss 用来建立状态空间模型，或者转换传递函数及 ZPK 系统为状态空间模型。ss 的书写格式为

sys = ss(A,B,C,D)

sys1 = ss(A,B,C,D,Ts)

式中，sys 用来描述连续系统；sys1 用来描述采样控制系统；Ts 为采样时间。

【例 11-2】 已知系统的状态方程为

$$x' = Ax + Bu$$
$$y = Cx + D$$

$$A = \begin{vmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -350 & -185 & -20 \end{vmatrix},\quad B = \begin{vmatrix} 0 \\ 110 \\ -1850 \end{vmatrix},\quad C = [1\quad 0\quad 0],\quad D = 0$$

求系统状态空间模型和传递函数表达式。

解:

```
≫A = [0 1 0;0 0 1; -350  -185  -20] %输入状态矩阵 A
A =
   0      1      0
   0      0      1
-350   -185    -20
≫B = [0;110; -1850];             %输入控制作用系数向量
≫C = [1 0 0];                    %输出向量
≫D = 0;                          %控制作用的 3 阶微商为零
≫sys = ss(A,B,C,D)               %计算状态模型
a =
```

```
        x1      x2       x3
   x1    0       1        0
   x2    0       0        1
   x3  -350    -185     -20
b =
              u1
   x1          0
   x2        110
   x3      -1850
c =
          x1   x2   x3
   y1      1    0    0
d =
          u1
   y1      0
Continuous-time model.              %该系统为时间连续模型
≫G = tf(sys)                         %转换成传递函数
Transfer function:
         110 s + 350
--------------------------
s^3 + 20 s^2 + 185 s + 350
≫step(G)                             %阶跃响应如图 11-2 所示
≫grid on
```

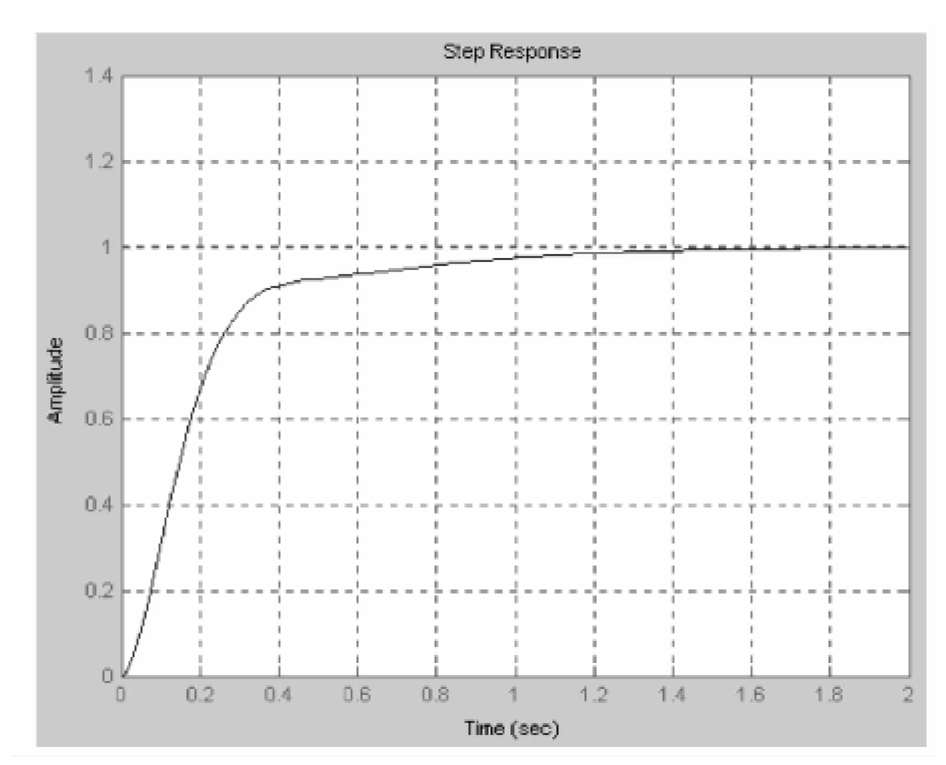


图 11-2　【例 11-2】状态空间模型的单位阶跃响应

11.3 传递函数的串联、并联和反馈连接

一个控制系统是由多个环节或模块组成，一般有调节对象、执行环节、放大运算环节、滤波环节和检测环节等。描述一个控制系统可以先从每个环节着手，分析它的输入输出特性，可以用物理分析的方法写出它的传递函数，也可以通过动态测试数据，来描述它的传递函数。当掌握了每一个环节的传递函数，就可以通过传递函数的合并规则描述整个系统的传递函数，分析它的稳定性和动态品质。对于单输入单输出系统，传递函数的合并不外乎串联，并联和反馈连接。

(1) 串联连接

MATLAB 是用 series 命令将两个 LTI 模型串联，两个模型要么都是连续系统，要么是用同一采样时间的离散系统。串联后的传递函数，是原来两个传递函数的乘积。Series 的书写格式为

sys = series(sys1,sys2)

式中，sys、sys1、sys2 是模型的传递函数表示形式或 ZPK 表示形式，并非为向量或矩阵表示形式。series 命令实际上是传递函数的乘法命令，它执行 sys = sys1 * sys2。

【例 11-3】 已知 $sys_1=\frac{s+1}{s^2+3s+1}$，$sys_2=\frac{2(s+2)}{(s+0.5)(s+5)}$

求串联传递函数的 ZPK 形式和传递函数形式及静态增益。

解：

```
≫sys1 = tf([1 1],[1 3 1])              %建立 sys1 的传递函数
Transfer function:
      s + 1
-----------------
s^2 + 3 s + 1
≫sys2 = zpk(-2,[-0.5; -5],2)          %建立 sys2 的 ZPK 形式
Zero/pole/gain:
    2 (s + 2)
------------------
(s + 0.5) (s + 5)
≫sys = series(sys1,sys2)               %串联连接
Zero/pole/gain:                         %串联后的 ZPK 形式
              2 (s + 2) (s + 1)
----------------------------------------
(s + 0.5) (s + 0.382) (s + 2.618) (s + 5)
≫systf = tf(sys)                       %串联后的传递函数形式
Transfer function:
            2 s^2 + 6 s + 4
-------------------------------------------
s^4 + 8.5 s^3 + 20 s^2 + 13 s + 2.5
```

```
≫gs = dcgain(sys)                      %串联后的静态增益
gs =
    1.6000
```

（2）并联连接

MATLAB 是用 parallel 命令将两个 LTI 模型并联，两个模型要么都是连续量系统，要么是用同一采样时间的离散系统。并联后的传递函数，是原来两个传递函数的相加。parallel 的书写格式为

sys = parallel(sys1, sys2)

式中，sys、sys1、sys2 是模型的传递函数表示形式或 zpk 表示形式，并非为向量或矩阵表示形式。parallel 命令实际上是传递函数的加法命令，它执行 sys = sys1 + sys2。

【例 11-4】 已知两个传递函数 $G_1=\frac{2s+3}{s^2+2s+5}$，$G_2=\frac{(s+3)}{(s+1)(s^2+3s+1)}$

求传递函数的并联连接，并求并联后的零点、极点、增益，以及在单位阶跃输入作用下的输出响应。

解：

```
≫G1 = tf([2 3],[1 2 5])                %建立传递函数 G1
Transfer function:
    2 s + 3
-----------------
s^2 + 2 s + 5
≫G2 = tf([1 3],conv([1 1],[1 3 1])) %建立传递函数 G2
Transfer function:
          s + 3
--------------------------
s^3 + 4 s^2 + 4 s + 1
≫G = parallel(G1,G2)                   %建立并联连接
Transfer function:
  2 s^4 + 12 s^3 + 25 s^2 + 25 s + 18
---------------------------------------
s^5 + 6 s^4 + 17 s^3 + 29 s^2 + 22 s + 5
≫[z,p,k] = zpkdata(G,'v')              %建立零点、极点和增益数据
z =
   -2.5826 + 0.4585i
   -2.5826 - 0.4585i
   -0.4174 + 1.0649i
   -0.4174 - 1.0649i
p =
   -2.6180
   -1.0000 + 2.0000i
```

```
  -1.0000 - 2.0000i
  -1.0000
  -0.3820
k =
     2
>>step(G,15),grid on          % 单位阶跃输入时的输出响应，时间设为15s，如图11-3所示
```

(3) 反馈连接

反馈是自动控制的精髓，有了反馈才能使控制目标达到给定精度，有了反馈才能降低参数变化和干扰的作用对系统输出的影响。在 MATLAB 中，是用 feedback 作为反馈连接的命令。假设前向通道的传递函数为 $G(s)$，反馈回路的传递函数为 $F(s)$，则 feedback 的书写格式为

sys = feedback(G,F,sign)

式中，sys 为反馈连接后的传递函数；sign 为反馈所取的符号，负反馈取 -1 或默认，正反馈取 +1 。

对于负反馈

$$sys = \frac{G(s)}{1 + G(s)F(s)}$$

对于正反馈则

$$sys = \frac{G(s)}{1 - G(s)F(s)}$$

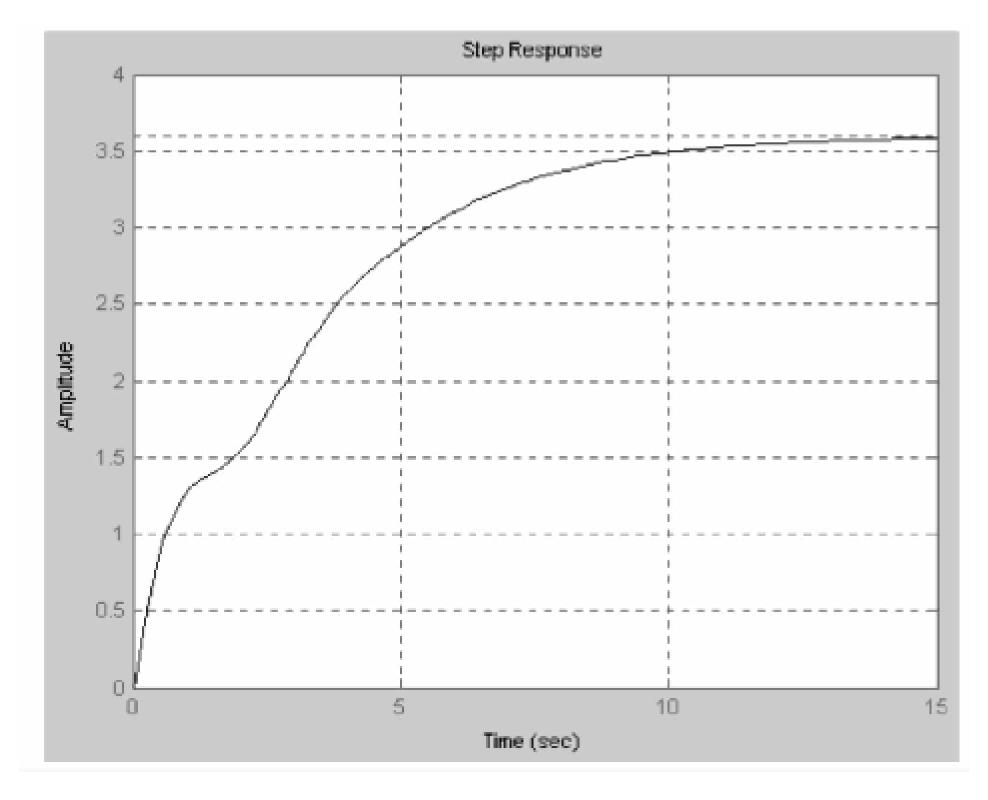


图 11-3　传递函数并联后单位阶跃输出响应

【例 11-5】 已知前向通道的传递函数 $G(s) = (10s + 100)/s^3 + 4s^2 + 50s + 1$，反馈通道的传递函数为 $F(s) = 1/(0.2s + 1)$，求反馈连接后的传递函数和单位阶跃响应。

解：

```
≫G = tf([10 100],[1 4 50 1])          %建立前向通道的传递函数
Transfer function:
        10 s + 100
------------------------------
s^3 + 4 s^2 + 50 s + 1
≫F = tf([1],[0.2,1])                  %建立反馈通道的传递函数
Transfer function:
      1
-------------------
0.2 s + 1
≫sys = feedback(G,F)                  %建立反馈连接
Transfer function:
                  2 s^2 + 30 s + 100
---------------------------------------------------
0.2 s^4 + 1.8 s^3 + 14 s^2 + 60.2 s + 101
≫step(sys)                            %单位阶跃响应如图 11-4
≫grid on                              %添加坐标栅格线
```

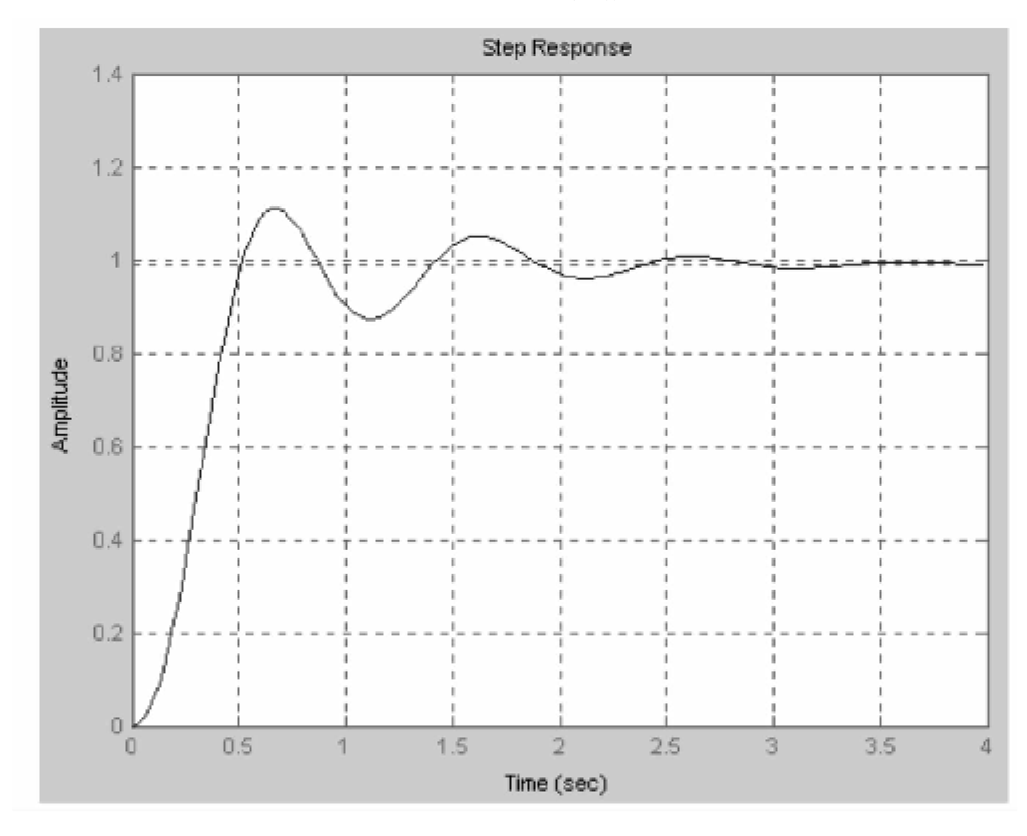


图 11-4　反馈连接后的单位阶跃响应

11.4 自动控制系统的稳定性

如果一个控制系统可以用 n 价线性定常微分方程式来描述，那么控制系统的稳定性就取决于微分方程式的稳定性。但是实际系统大多带有非线性，由于非线性问题的复杂性，这给分析带来一定的难度。如果状态变量的变化范围不大，则可以认为系统的参数都是定常的，输入与输出的关系都是线性的。

对于线性定常微分方程的稳定性，只要判别特征方程式的根是否全在复数平面的左半面上，如果是，那么系统就是稳定的。不少数学家为控制系统的稳定性分析做出贡献，如劳斯判据、赫尔维茨行列式、李亚普诺夫稳定性准则、奈奎斯特稳定判据等。MATLAB 的稳定性分析可采用零极点分布图（pzmap）、状态矩阵的特征值分解和奈奎斯特（nyquist）稳定判据等。作者则用赫尔维茨行列式编制程序，使计算更为方便。

1. 用零极点分布图 pzmap 分析系统稳定性

pzmap 已经在 11.1 节中使用过。它的书写格式为

```
pzmap(sys)
[p,z] = pzmap(sys)
```

pzmap(sys)给出零极点分布图。sys 为 LTI 的模型，它可以是传递函数、ZPK 模式，也可以是状态空间模型，所以使用极为方便。

[p,z] = pzmap(sys)，只给出极点和零点的向量，不提供零点极点分布图。

【例 11-6】 已知 3 阶系统的状态空间数据 $A=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & 1\\ -1 & -3 & -2\end{bmatrix}$，$B=[0\quad 0.2\quad 0.6]$，$C=[1\quad 0\quad 0]$，$D=0$，求零点、极点及其传递函数。分析它的稳定性。

解：

```
>>A = [0 1 0;0 0 1; -1  -3  -2]        %输入状态空间参数
A =
     0     1     0
     0     0     1
    -1    -3    -2
>>B = [0;0.2;0.6];
>>C = [1 0 0];D = 0;
>>sys = ss(A,B,C,D);                   %建立状态空间模型
>>[p,z] = pzmap(sys)                   %计算零点、极点向量
p =
  -0.4302
  -0.7849 + 1.3071i
  -0.7849 - 1.3071i
z =
  -5.0000
```

由于极点全在左半复平面，所以系统是稳定的

```
≫G = tf(sys)                          %状态空间模型转换成传递函数
Transfer function:
      0.2 s + 1
---------------------------
s^3 + 2 s^2 + 3 s + 1
```

2. 用状态空间的特征值分析稳定性

采用状态空间表示法，只要分析系统的状态矩阵的特征值就可以了。因为状态矩阵的特征值与传递函数的特征根完全相同。若特征值为 λ，则

$$|\lambda \boldsymbol{I} - \boldsymbol{A}| = 0$$

的方程式即为特征方程式

$$\lambda^n + a_1\lambda^{n-1} + \cdots + a_{n-1}\lambda + a_n = 0$$

只要状态矩阵的特征值全在左半复平面，则系统是稳定的。

【例 11-7】 已知系统的状态矩阵为 $\boldsymbol{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -1 & -3 & -2 \end{bmatrix}$，求解系统的稳定性。

解:

```
≫A = [0 1 0;0 0 1;-1 -3 -2]        %输入状态矩阵
A =
    0    1    0
    0    0    1
   -1   -3   -2
≫eig(A)                              %状态矩阵 A 的特征值，与上例极点向量完全一致。
                                      因此系统是稳定的
ans =
   -0.4302
   -0.7849 + 1.3071i
   -0.7849 - 1.3071i
```

【例 11-8】 今有非标准型的 4 阶系统状态矩阵 $\boldsymbol{A} = \begin{bmatrix} -2 & 2 & 0 & 0 \\ 0 & -4 & 4 & 0 \\ 0 & 0 & -5 & 5 \\ 0 & 0 & 0 & -10 \end{bmatrix}$，判别它的稳定性。

解:

```
≫A = [-2 2 0 0;0 -4 4 0;0 0 -5 5;0 0 0 -10]     %输入系统状态矩阵
A =
     -2      2      0      0
      0     -4      4      0
      0      0     -5      5
      0      0      0    -10
```

```
≫eig(A)                                %计算特征值，全为负实部，因此系统稳定
ans =
     -2
     -4
     -5
     -10
```

3．用赫尔维茨行列式分析系统的稳定性

赫尔维茨行列式是由19世纪德国数学家赫尔维茨（Hurwitz）为分析控制系统的稳定性而建立的准则。该准则认为假如一个 n 阶线性定常系统的特征方程式为

$$a_n s^n + a_{n-1} s^{n-1} + \cdots + a_1 s + a_0 = 0 \tag{11-1}$$

那么赫尔维茨行列式由 n 阶方阵组成为

$$\begin{vmatrix} a_{n-1} & a_{n-3} & a_{n-5} & a_{n-7} & \cdots & 0 \\ a_n & a_{n-2} & a_{n-4} & a_{n-6} & \cdots & 0 \\ 0 & a_{n-1} & a_{n-3} & a_{n-5} & \cdots & 0 \\ 0 & a_n & a_{n-2} & a_{n-4} & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & a_4 & a_2 & a_0 \end{vmatrix} \tag{11-2}$$

赫维尔茨行列式是以如下原则排列的：

1）主对角线元素，从 a_{n-1} 排起，直到常数项 a_0；

2）主对角线以上元素，每隔一行，下标数减1；

3）主对角线以下元素，每隔一行，下标数加1；

4）矩阵元素的下标，对大于 n 或小于零的元素，则以零替代。

赫尔维茨准则判别系统稳定性是要求所有的特征多项式系数全为正，并且沿着主对角线的从 $2\sim n-1$ 阶的赫尔维茨子行列式都大于零，n 阶行列式亦大于零，则系统稳定，否则系统是不稳定的。

例如一个3阶系统，它的特征方程式为

$$a_3 s^3 + a_2 s^2 + a_1 s + a_0 = 0 \tag{11-3}$$

则它的赫尔维茨行列式为

$$\begin{vmatrix} a_2 & a_0 & 0 \\ a_3 & a_1 & 0 \\ 0 & a_2 & a_0 \end{vmatrix} \tag{11-4}$$

系统稳定条件为

$$a_3、a_2、a_1、a_0 > 0 \tag{11-5}$$

$$a_1、a_2 - a_0 a_3 > 0 \tag{11-6}$$

$$a_0(a_1 a_2 - a_0 a_3) > 0 \tag{11-7}$$

所以在满足正系数的条件下，满足式（11-6），系统是稳定的。但是随着阶次的提高，分析计算越来越困难。因此考虑用编程的方法，当输入特征方程式系数时，赫尔维茨行列式能自动进行计算，并判断系统是否稳定。

下面的程序是用 MATLAB 的 M 文件编写的，当特征多项式的系数是已知时，计算 2～n 阶赫尔维茨行列式，以判别系统是否稳定。

```
% This program for calculate the determinant of Hurwitz     % 程序标题
clear                                                       % 清内存变量
V = input('input a vector of polynormial coefficient \ n')  % 输入特征多项式系数向量的提示
[m,n] = size(V);                                            % 测试向量大小
n = n - 1                                                   % 系统阶次为向量 V 的列数减 1
if m ~ = 1                                                  % 若行数不等于 1，则显示出错
    error('The input must be a vector')
end
A = zeros(1,3* n - 2);                                      % 根据阶次设置扩充的零向量 A，为建立赫尔维茨行列式做准备
A(n - 1:2* n - 1) = V;                                      % 把多项式系数向量 V，填入零向量 A
for i = 1:n                                                 % 对向量 A，依次向方阵 S 写入。第一行取向量 A 的 1： n - 1 个元素，下一行向取向量 A 的 3：n - 4 元素，依此类推
    S(i,1:n) = A(2* i - 1:2* i + n - 2);
end
S = flipud(S');                                             % 将 S 转置成 S′，再将 S′上、下翻转，其中 flipud 为矩阵上下翻转函数。翻转后，再赋予 S，即为赫尔维茨行列式
for i = 2:n
    D(i) = det(S(1:i,1:i));                                 % 计算 2 ~ n 赫尔维茨行列式
end
D(2:n)'                                                     % 显示赫尔维茨行列式
if D(2:n) > 0                                               % 假若赫尔维茨行列式全大于零
    disp('system is stabilization')                         % 显示系统是稳定的
else
    disp('system is unstabilization')                       % 否则是不稳定的
end
```

下面举例子来说明。

【例 11-9】 设已知一个系统的特征方程式为

$$s^4 + 2s^3 + 3.5s^2 + 2.7s + 1 = 0 \tag{11-8}$$

试用赫尔维茨行列式来判断其稳定性。

在命令窗口输入如下程序：

解:

```
≫syms s                                        %设 s 为符号变量
≫V = s^4 + 2* s^3 + 3.5* s^2 + 2.7* s + 1;    %设置特征方程式 V
≫V = sym2poly(V)                               %将特征方程式转换成系数向量
V =
    1.0000    2.0000    3.5000    2.7000    1.0000
≫Hurwitz                                        %调用 M 文件 Hurwitz (程序名，即计算赫尔
                                                 维茨行列式程序)
input a vector of polynormial coefficient       %提示：输入多项式系数向量
[1 2 3.5 2.7 1]
V =                                              %程序响应
    1.0000    2.0000    3.5000    2.7000    1.0000
S =                                              %显示赫尔维茨行列式
    2.0000  2.7000       0       0
    1.0000  3.5000  1.0000       0
         0  2.0000  2.7000       0
         0  1.0000  3.5000  1.0000
ans =                                            %各阶赫尔维茨行列式，全大于零
    4.3000                                       %2 阶行列式
    7.6100                                       %3 阶行列式
    7.6100                                       %4 阶行列式
system is stabilization                          %显示系统是稳定的
```

【例 11-10】 已知 5 阶系统特征方程式的系数向量为 $\boldsymbol{V}$ = [1 1 4 4 2 1]，试用赫尔维茨行列式来判别其稳定性。

解:

```
≫Hurwitz                                        %调用 M 文件 Hurwitz (程序名，即计算赫尔
                                                 维茨行列式程序)
input a vector of polynormial coefficient       %提示：输入系数向量
[1 1 4 4 2 1]
V =                                              %程序响应
    1    1    4    4    2    1
S =                                              %显示赫尔维茨行列式
    1    4    1    0    0
    1    4    2    0    0
    0    1    4    1    0
    0    1    4    2    0
    0    0    1    4    1
ans =                                            %各阶赫尔维茨行列式显示在默认向量 ans 内，
                                                 全不大于零
```

```
   0
 -1
 -1
 -1
system is unstabilization                 %该系统是不稳定系统
```

4. 用符号变量求解赫尔维茨行列式

在设计控制系统前，往往只知道调节对象的参数，所以系统的特征方程式中经常含有调节器的参数或称之为参变量。因此用符号变量求解赫尔维茨行列式是有必要的。对上述程序稍加改动即可用于解符号变量的赫尔维茨行列式。其程序如下：

```
%This program for solution the determinant of Hurwitz by symbol variable
                                          %程序标题
clear                                     %清内存变量
V = input('input a vector of polynormial coefficient by symbol variable \n')
                                          %输入特征多项式符号变量向量
[m,n] = size(V);                          %测试向量大小
n = n - 1;                                %系统阶次为向量 V 的列数减 1
if m ~ = 1                                %若行数不等于 1，则显示出错
    error('The input must be a vector')
end
A = zeros(1,3*n-2);                       %根据阶次设置扩充的零向量 A
A = sym(A);                               %转换零向量 A 为符号变量，sym 为符号变
                                            量转换函数
A(n-1:2*n-1) = V;                         %把多项式符号变量向量 V，填入零向量 A
for i = 1:n                               %对向量 A，依次向方阵 S 填参数。第一行取
                                            向量 A 的 1：n-1 个元素，下一行取向量 A
                                            的 3：n-4 元素，依此类推
    S(i,1:n) = A(2*i-1:2*i+n-2);
end
S = flipud(S');                           %将 S 转置成 S'，再将 S'上、下翻转，再赋予
                                            S，即为赫尔维茨行
for i = 2:n                               %计算 2~n 阶行列式
    D(i) = det(S(1:i,1:i));
end
D = D(2:n)'                               %显示 2~n 阶行列式，以行形式排列
```

下面举例子予以说明。

【例 11-11】 今有一 4 阶系统，其特征方程式为

$$a_4 s^4 + a_3 s^3 + a_2 s^2 + a_1 s + a_0 = 0 \tag{11-9}$$

求其赫尔维茨行列式。

解：

在命令窗口输入如下程序：

```
≫Hurwitz_sym                                    %调用计算赫尔维茨行列式(符号变量)函数
input a vector of polynormial coefficient by symbol variable
                                                %程序响应
sym('[a4 a3 a2 a1 a0]')                         %输入多项式系数的符号变量向量，sym 为符
                                                  号变量转换函数
V =
[ a4, a3, a2, a1, a0]
D =                                             %赫尔维茨行列式计算结果
[                    a3* a2 - a1* a4]           %2 阶子行列式
[        a3* a2* a1 - a0* a3^2 - a4* a1^2]      %3 阶子行列式
[a3* a2* a1 * a0 - a3^2* a0^2 - a4* a1^2* a0]   %4 阶行列式
≫D1 = simple(D)                                 %化简，simple 为化简函数
D1 =
[                    a3* a2 - a1* a4]
[        a3* a2* a1 - a0* a3^2 - a4* a1^2]
[a0* (a3* a2* a1 - a0* a3^2 - a4* a1^2)]
```

由结果可知，4 阶系统在系数为正时的稳定条件为

$$a_3 a_2 - a_1 a_4 > 0 \tag{11-10}$$

$$a_3 a_2 a_1 - a_0 a_3^2 - a_4 a_1^2 > 0 \tag{11-11}$$

将 n 阶赫尔维茨行列式用块矩阵展开，则

$$\boldsymbol{D}(n) = a_0 \boldsymbol{D}(n-1) - \mathrm{zeros}(n-1,1)\boldsymbol{D}(n,1:n-1) = a_0 \boldsymbol{D}(n-1) \tag{11-12}$$

其中zeros$(n-1,1)$为 1 列 $n-1$ 行的零向量；$\boldsymbol{D}(n,1:n-1)$为非零行向量 $(n-1$ 列)；$\boldsymbol{D}(n-1)$为 $n-1$ 阶子行列式。

所以判断系统是否稳定，只要分析在主对角线上各阶子行列式是否大于零即可。

【例 11-12】 计算图 11-5 所示系统的稳定条件。其中前向第一方块为 PI 调节器，k_r 为比例系数，t_3 为积分时间常数。前向第二个方块为调节对象，k 为增益，t_1 为惯性环节时间常数，t_2 为积分时间常数，k_f 为反馈通道传递系数。假设 $k^* k_f = 1$。k_r、t_3 为待求参数。

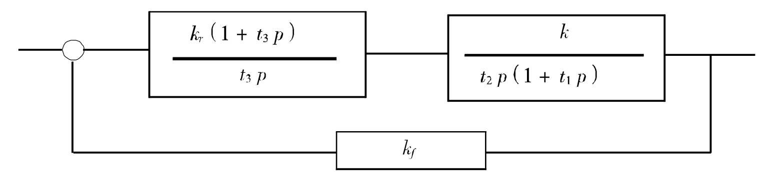


图 11-5 某系统框图

解：该系统的闭环传递函数为

$$G(p) = \frac{k_r k(1 + t_3 p)}{t_3 t_2 p^2 (1 + t_1 p) + kr \cdot k \cdot k_f (1 + t_3 \cdot p)} \tag{11-13}$$

其特征方程式为

$$V(p) = t_1 t_2 t_3 p^3 + t_2 t_3 p^2 + k_r t_3 p + k_r \tag{11-14}$$

将式 (11-14) 的符号多项式系数输入到计算赫尔维茨行列式程序中，则得到如下：

```
≫hurwitz_sym                                                %调用程序
input a vector of polynormial coefficient by symbol variable %输入特征多项式系数
sym('[t1*t2*t3 t2*t3 kr*t3 kr]')
V =
[ t1*t2*t3,   t2*t3,   kr*t3,   kr]
D =                                                         %2或3阶赫尔维茨行列式
[      t2*t3^2*kr - kr*t1*t2*t3]
[t2*t3^2*kr^2 - t1*t2*t3*kr^2]
≫D1 = simple(D)                                             %化简后的2或3阶赫尔维茨行列式
D1 =
[    - t2*t3*kr*(- t3 + t1)]
[- t2*t3*kr^2*(- t3 + t1)]
```

为了使 D_1 各行大于零，则 $t_3 > t_1$，而稳定性与 k_r 无关。但超调量是与 k_r 有关，一般情况下，k_r 增大，超调量也增加。为了使系统有足够的稳定裕量，通常选择 t_3 等于2~5倍的 t_1。

t_3 的数值过大，将会使系统的单位阶跃响应过程变慢。

11.5 根轨迹图的绘制

LTI系统的基本特性是由描述系统的传递函数或状态方程决定的，而传递函数的特性，则主要取决于特征方程式的极点分布。根轨迹图的基本思想是，使 $|1 + k(s)| = 0$ 时静态增益 k 在［0，inf］变化时，闭环极点相应变化的规律。式中 $k(s)$ 为系统开环传递函数。有了根轨迹图，设计者可以选择闭环极点的位置，选择系统相应的参数和静态增益，以达到满意的动态性能。

绘制根轨迹图，在以往是相当费时的。有了MATLAB，使绘制根轨迹图变得容易。MATLAB的根轨迹图绘制函数为rlocus，它的书写格式为

```
rlocus(sys)
rlocus(sys,k)
[r,k] = rlocus(sys)
r = rlocus(sys,k)
```

rlocus(sys)绘制由系统模型sys描述的根轨迹图。sys可以是传递函数，也可以是系统状态空间模型。

rlocus(sys,k)绘制由设定向量k所绘制的闭环极点图。

[r,k] = rlocus(sys)则输出闭环极点r和开环增益k的向量，而不绘制根轨迹图。

r = rlocus(sys,k)则输出闭环极点r，对应设定的向量k，也不绘制根轨迹图。

与rlocus函数相对应的函数是rlocfind。它是从根轨迹图上寻找闭环极点相对应的增益。

【例11-13】 已知开环传递函数的分子num = $2s + 1$，分母den = $s(s^2 + 3s + 2)$，求根轨迹图，保持图形，并添加 $k = 2, 4, 6, \cdots, 12$ 时的极点分布图。

解：

```
≫num = [2 1];                        %输入分子多项式系数
≫den = conv([1 0],[1 3 2]);          %输入分母多项式系数
≫G = tf(num,den)                     %创建传递函数
Transfer function:
    2 s + 1
----------------------
s^3 + 3 s^2 + 2 s
≫rlocus(G)                           %绘制根轨迹图，如图 11-6 所示
≫hold on                             %图形保持
≫k = 2:2:12;                         %设置向量 k
≫rlocus(G,k)                         %绘制向量 k 时的极点
```

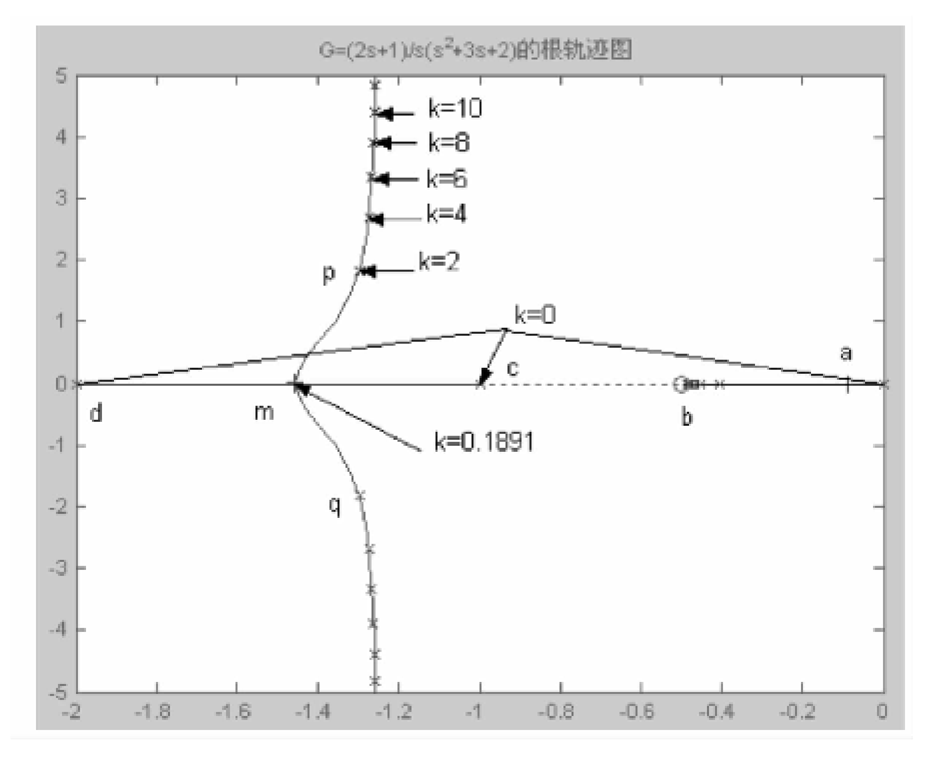


图 11-6　$G(s)=(2s+1)/[s(s^2+3s+2)]$ 的根轨迹图

根轨迹图的分析

（1）当 k 趋近零时，闭环极点分布为 −0、−1、−2。

（2）当 k 增大时，极点 a 向零点 b 靠近。极点 c、d 向实轴上 m 点靠近，再向复极点 p、q 发展。

（3）系统是稳定的，因为闭环极点始终在左半复平面。但随着 k 值增大，系统阻尼系数逐渐下降，因而振荡频率增加，超调量增加。

【例 11-14】　已知调节对象传递函数 $G=50/(0.08s+1)/(0.12s+1)/(2s+1)$，反馈回路的反馈传递函数为 $H=1/(0.02s+1)$，为使系统的稳定，在调节对象前串联一个 PI 调节器 R，R 的传递函数为 $2k_p \cdot s/(2s+1)$，如图 11-7 所示。为使系统阻尼系数为 0.6，求调节器传递系数 k_p。

解：

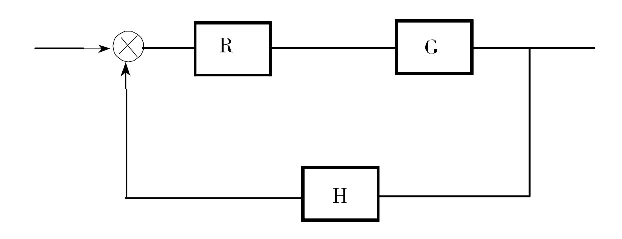


图 11-7 某系统框图

设 PI 调节器的比例系数 k_p 暂定为 1。在命令窗口中输入如下程序：

```
≫R = tf([2 1],[2 0])                          % PI 调节器的传递函数,kp = 1
Transfer function:
2 s + 1
---------
  2 s
≫den = conv(conv([0.05 1],[0.12 1]),[2 1])   % 调节对象 G 的分母多项式
den =
    0.0120     0.3460     2.1700   1.0000
≫G = tf(50,den)                               % G 的传递函数
Transfer function:
                         50
-----------------------------------------
0.012 s^3 + 0.346 s^2 + 2.17 s + 1
≫G0 = series(R,G)                             % 调节器与调节对象相串联
Transfer function:
                   100 s + 50
-----------------------------------------------
0.024 s^4 + 0.692 s^3 + 4.34 s^2 + 2 s
≫H = tf([1],[0.02 1]);                        % 反馈回路传递函数
≫G0 = series(G0,H)                            % 与反馈回路传递函数相串联
Transfer function:
                             100 s + 50
-------------------------------------------------------------------
0.00048 s^5 + 0.03784 s^4 + 0.7788 s^3 + 4.38 s^2 + 2 s
≫rlocus(G0)                                   % 绘制根轨迹图，如图 11-8 所示
≫axis([-30,0,-40,40])                         % 坐标轴设置。接着，在图 11-8 上绘制主极
                                                点的阻尼比为 0.6 的直线与根轨迹曲线相
                                                交
≫rlocfind(G0)                                 % 求交点处的闭环增益
Select a point in the graphics window          %要求在图形窗口选点，即阻尼比为 0.6 的辅
                                                助线与根轨迹曲线的交点
selected _ point =                             % 交点处极点坐标
```

```
    -2.8081 + 3.8509i
ans =                                   % 闭环增益,即允许的 kp 值
     0.1285
kp = ans;                               % 选择 kp 的值
```

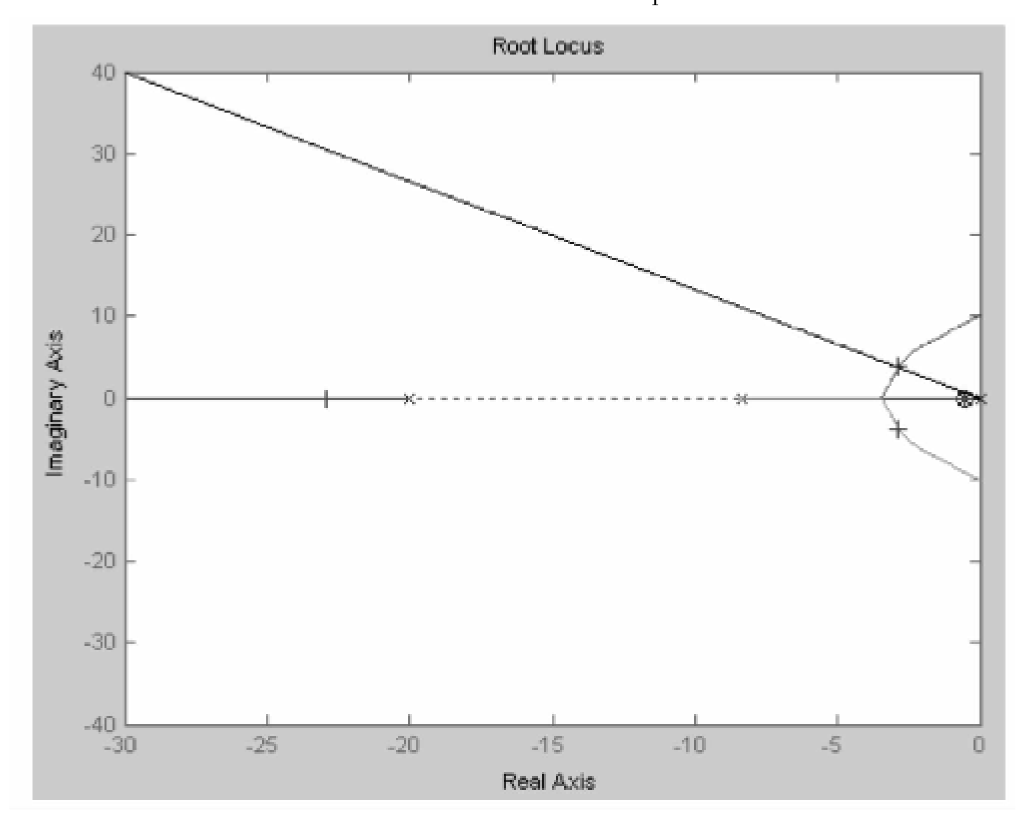


图 11-8　G0 的根轨迹图

```
≫G0 = series(G, kp* R)                  % 比例系数确定后的前向传递函数
Transfer function:
              12.85 s + 6.426
-----------------------------------------------
0.024 s^4 + 0.692 s^3 + 4.34 s^2 + 2 s
≫Gc = feedback(G0, H)                   % 闭环传递函数
Transfer function:
                    0.2571 s^2 + 12.98 s + 6.426
---------------------------------------------------------------------------------
0.00048 s^5 + 0.03784 s^4 + 0.7788 s^3 + 4.38 s^2 + 14.85 s + 6.426
≫zpk(Gc)                                % 传递函数的 ZPK 形式
Zero/pole/gain:
               535.5233 (s + 50) (s + 0.5)
    ---------------------------------------------------------------
(s + 49.56) (s + 22.95) (s + 0.5) (s^2  + 5.826s + 23.54)
step(Gc)                                % 绘制单位阶跃输入时的输出响应, 如图
                                          11-9所示
```

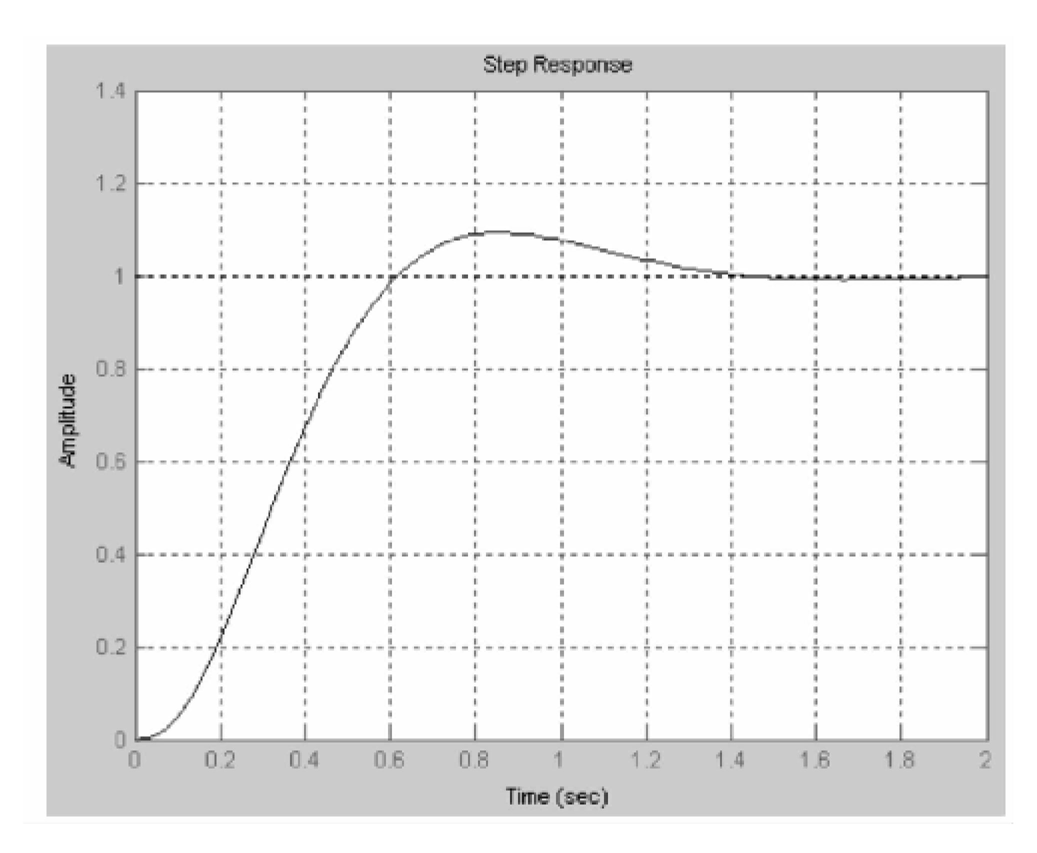


图 11-9 用设定系统阻尼比来确定调节器比例系数

```
≫max(step(Gc))                         %最大输出
ans =
    1.0921
delta = (ans - 1) = 9.21%              %超调量
```

【例 11-15】 已知某系统的状态空间模型 $A=\begin{bmatrix}0 & 1 & 0\\ -4 & -1 & 1\\ 0 & 0 & -20\end{bmatrix}$，$B=[0\quad 0\quad 20]$，$C=[1\quad 0\quad 0]$。$D=[0]$，求根轨迹图及其传递函数。

解：

```
≫A = [0 1 0; -4 -1 1;0 0 -20];          %输入状态空间模型参数
≫B = [0;0;20];
≫C = [1 0 0];
≫D = [0];
≫rlocus(A,B,C,D)                       %绘制根轨迹图，如图 11-10 所示
≫G = tf(ss(A,B,C,D))                   %转换成传递函数
Transfer function:
              20
-----------------------------------
s^3 + 21 s^2 + 24 s + 80
```

因此得到传递函数为

$$\frac{20}{s^3+21s^2+24s+80}$$

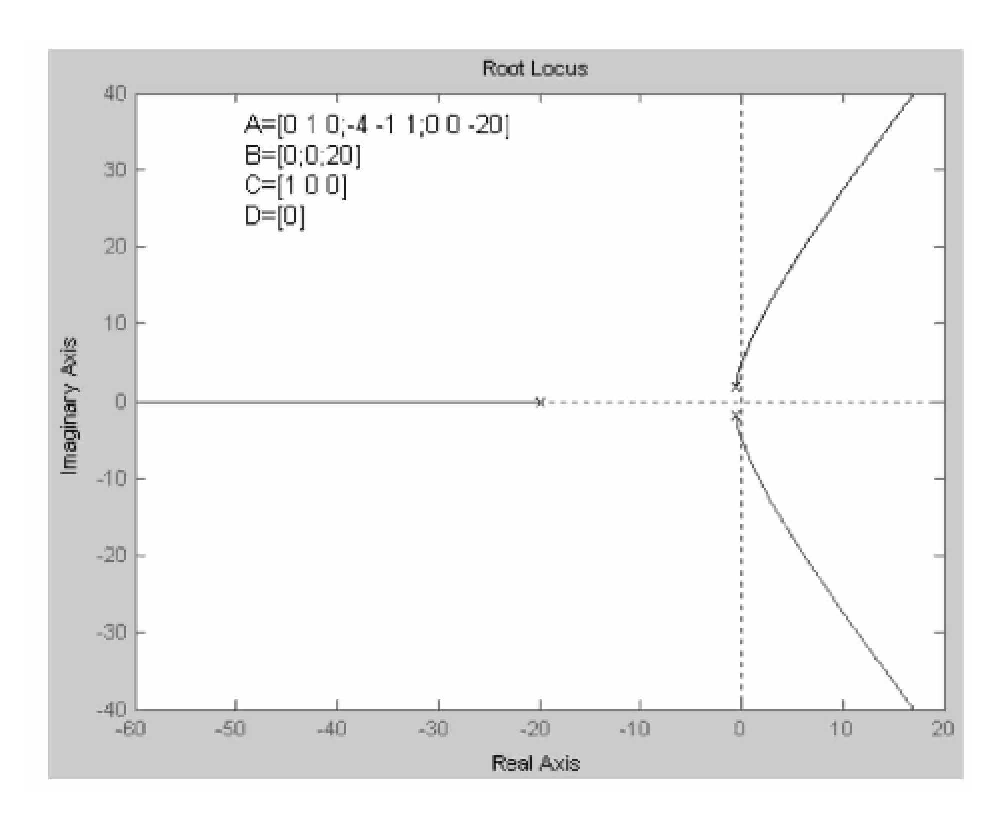


图 11-10 由状态空间模型表示的根轨迹图

11.6 博德图、尼柯尔斯图和奈奎斯特图的绘制

描述自动控制系统存在两种分析法，一种是时域分析法，另一种是频域分析法。使用传递函数和状态空间是属于时域分析法而对系统输入正弦信号，观察它的稳态输出响应的方法称为频域分析法。时域分析法与频域分析法是互相独立又互为补充的。频域分析法的优点之一是可以通过对系统的测试而获得系统的频率响应曲线，而不必通过推导系统的数学模型。频域分析法是由奈奎斯特、博德、尼柯尔斯等学者发展起来的。

博德图是由系统的频率响应幅值图和频率响应相角图构成。这二幅图都是相对于频率的对数坐标绘制的。MATLAB 中绘制博德图的函数为 bode，它的书写格式为

```
bode(num,den)
bode(num,den,w)
bode(A,B,C,D)
bode(A,B,C,D,w)
bode(sys)
[mag,phase]=bode(num,den,w)
```

式中，num、den 为传递函数的分子、分母的系数多项式。w 为指定的频率向量。A、B、C、D 为系统的状态空间矩阵。mag、phase 为指定频率下的幅值和相角。sys 为系统模型表达式。在绘制博德图过程中，用来计算幅值稳定裕量 Gm，相角稳定裕量 Pm，幅值截止频率 Wcg，相角截止频率 Wcp 的函数 margin 是很有用的。它的书写式为

```
[Gm,Pm,Wcg,Wcp]=margin(sys)
[Gm,Pm,Wcg,Wcp]=margin(mag,phase,w)
margin(sys)
```

表达式[Gm，Pm，Wcg，Wcp] = margin(sys)，用来计算系统模型为 sys 的稳定裕量。

表达式[Gm，Pm，Wcg，Wcp] = margin(mag，phase，w)是用已知在博德图上的幅值(mag)向量、相角(phase)向量和角频率(w)向量，w 估算稳定裕量。

表达式 margin(sys)，则用来绘制博德图，并在图上显示幅值稳定裕量，相角稳定裕量等数据。

尼柯尔斯图也是用来绘制系统的对数幅值与相角的关系曲线。它的书写格式如下：

```
nichols(num，den)
nichols (num，den，w)
nichols (A，B，C，D)
nichols (A，B，C，D，w)
nichols (sys)
[mag，phase] = nichols (num，den，w)
```

尼柯尔斯图与博德图的区别是它把对数幅值和相角的曲线绘在一张复平面上。

奈奎斯特图则是绘制系统的幅相特性曲线在复数平面上，它的书写格式为

```
nyquist(num，den)
nyquist(num，den，w)
nyquist(A，B，C，D)
nyquist(A，B，C，D，w)
nyquist(sys)
[mag，phase] = nyquist (num，den，w)
```

【例 11-16】 考虑下列传递函数

$$G(s) = \frac{10}{s^2 + 4s + 10}$$

求其博德图、尼柯尔斯图和奈奎斯特图。

解：

```
>>num = 10; den = [1 4 10];          %传递函数的分子和分母
>>bode(num，den)                     %绘制博德图，如图 11-11 所示
>>grid on                            %添加坐标线
>>figure                             %建立新的图形窗口
>>nichols(num，den)                  %绘制尼柯尔斯图，如图 11-12 所示
>>grid on                            %添加坐标线
>>figure                             %建立新的图形窗口
>>nyquist(num，den)                  %绘制奈奎斯特图，如图 11-13 所示
>>grid on                            %添加坐标线
```

【例 11-17】 已知开环传递函为 $G = \dfrac{240(s+1)}{s^2(s+5)(s+15)}$，反馈传递函数 $H = 1$，用博德图分析它的稳定性、幅值截止频率和相位截止频率。计算它的谐振峰值和频率，并绘制单位阶跃输入时的输出响应。

解：

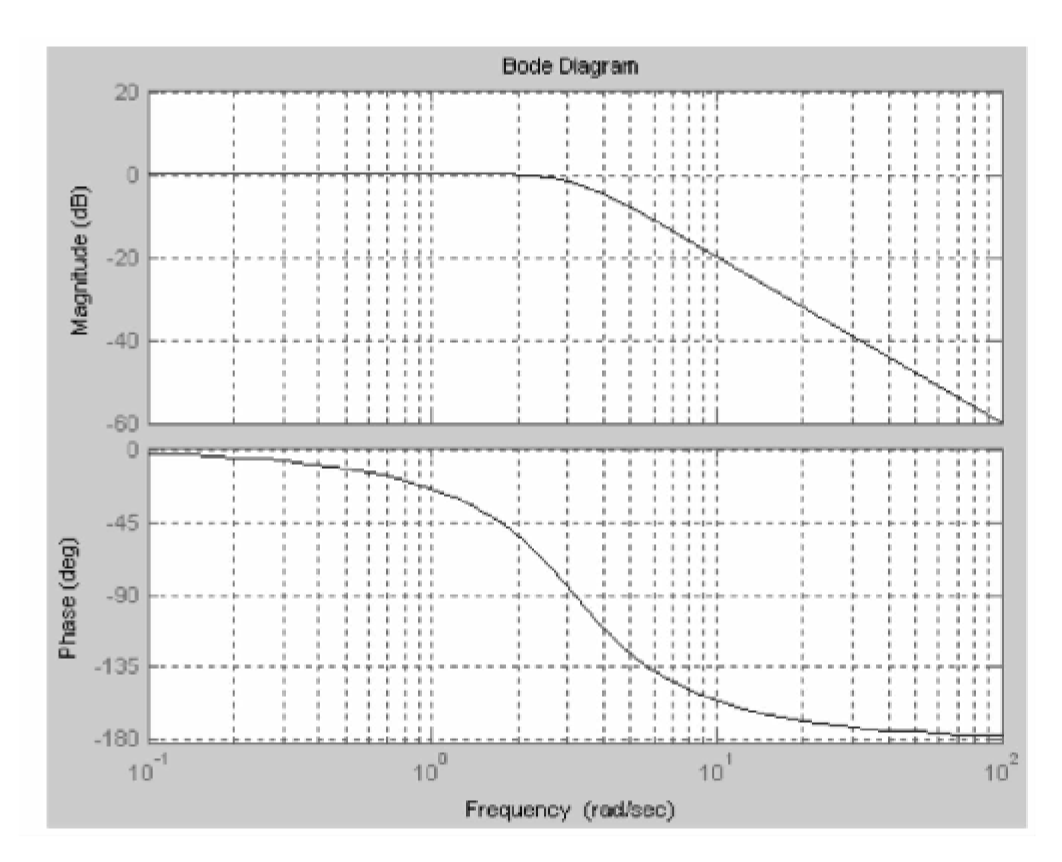


图 11-11 *G*（*s*）的博德图

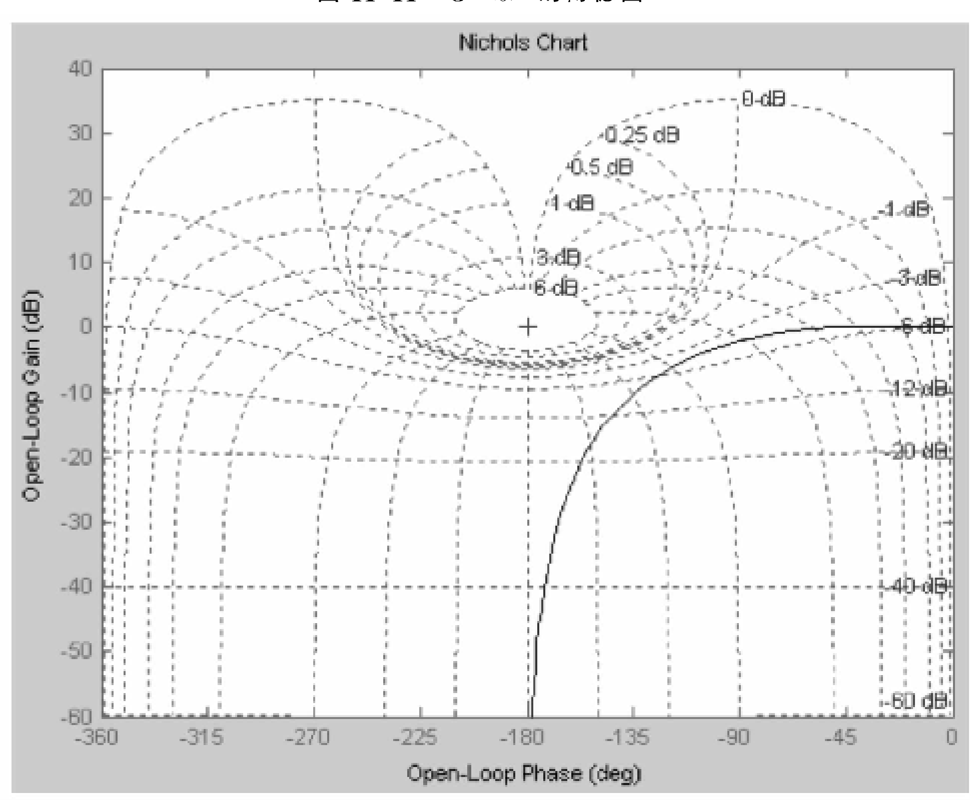


图 11-12 *G*（*s*）的尼柯尔斯图

```
≫G = tf([240,240],conv([1 0 0],[1 20 75]))        %创建传递函数
Transfer function:
        240 s + 240
```

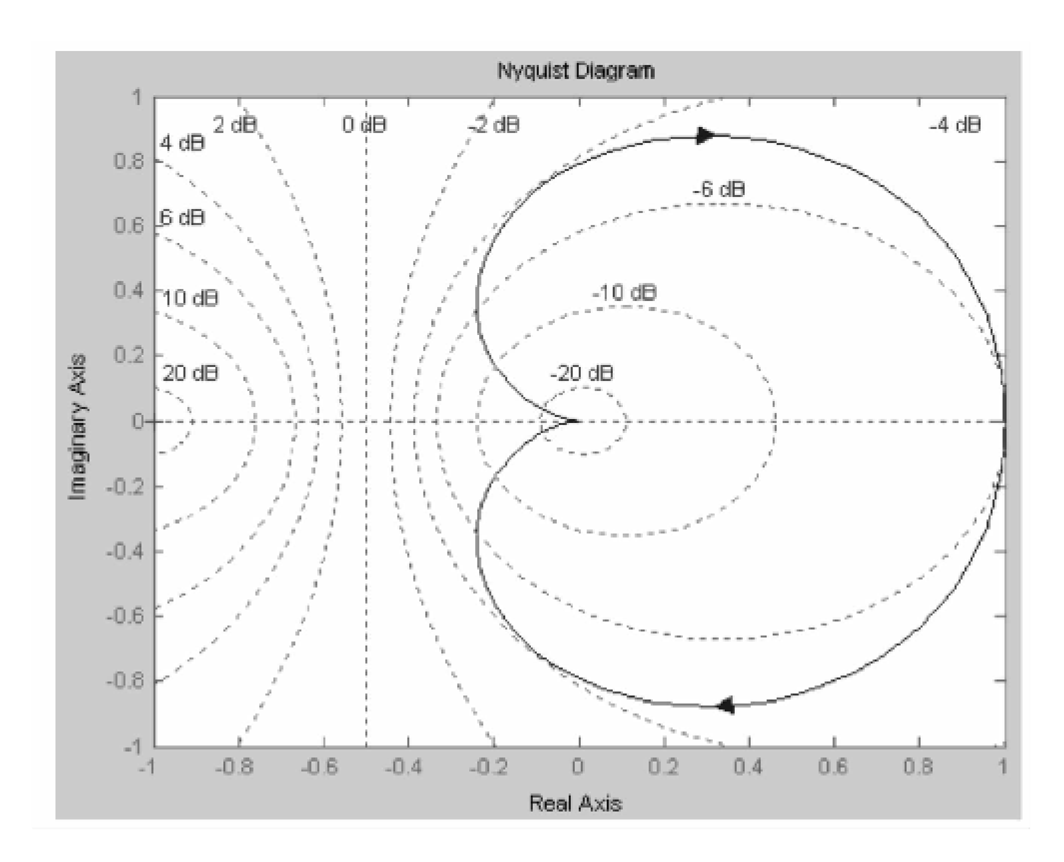


图 11-13　*G*（*s*）的奈奎斯特图

```
-------------------------------
s^4 + 20 s^3 + 75 s^2
≫bode(G)                    %绘制博德图，如图 11-14 所示，从图中可知相角
                            裕量为 30deg，截止频率为 3rad/s，相位截止频率
                            为 7.5rad/s，系统稳定
≫grid on                    %添加坐标网格线
≫Gc = feedback(G,[1])       %建立闭环传递函数
Transfer function:
              240 s + 240
-------------------------------------------
s^4 + 20 s^3 + 75 s^2 + 240 s + 240
≫w = 1:0.02:15;             %设置角频率向量
≫[mag,ph] = bode(Gc,w);     %计算闭环频率特性的幅值和相角特性
≫[maxmag,wi] = max(mag)     %计算幅值特性的最大值
maxmag =                    %谐振峰值
      1.9325
wi =                        %对应角频率向量的下标数
      92
≫wc = w(wi)                 %查找相应的角频率
wc =                        %谐振频率
      2.8200
```

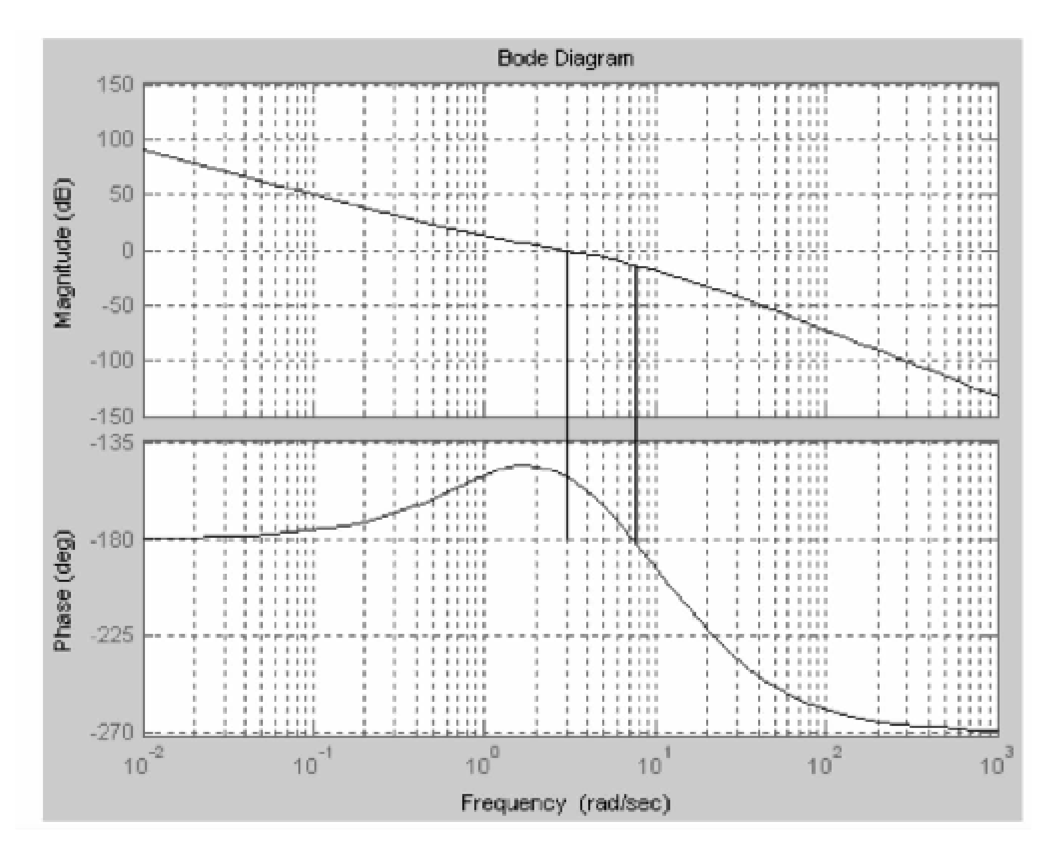


图 11-14 G 的博德图

```
≫step(Gc)                    %绘制单位阶跃响应，如图 11-15 所示
≫grid on
```

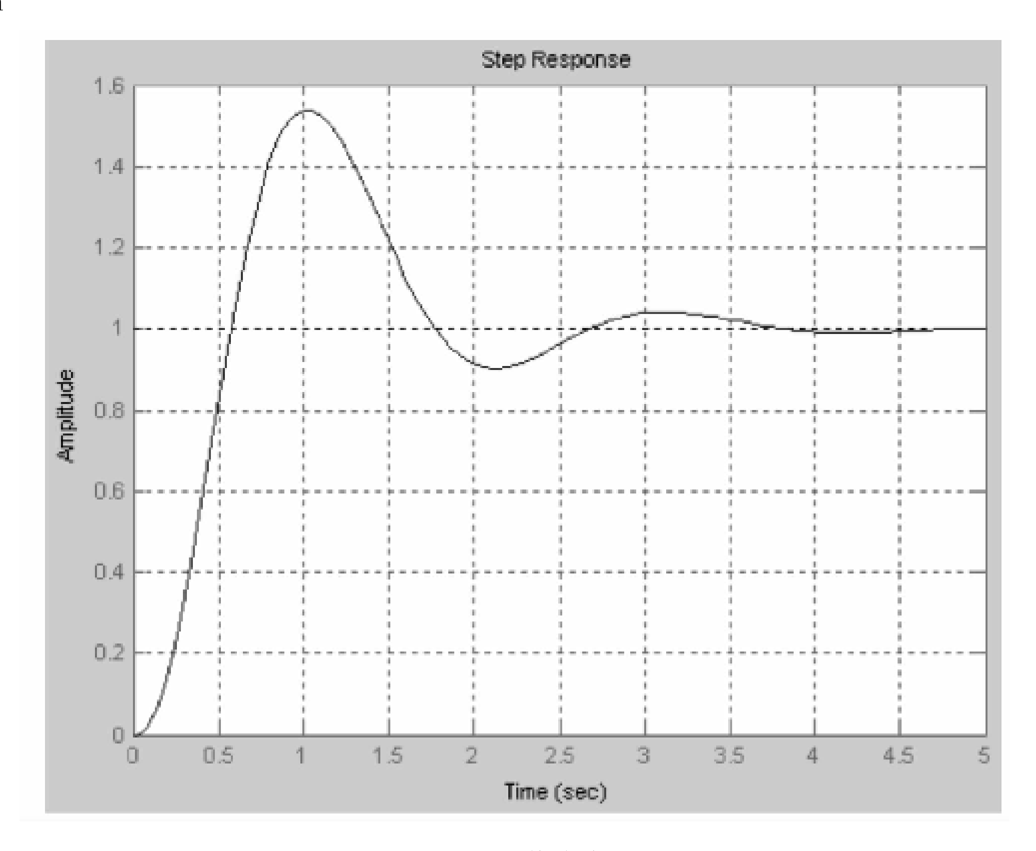


图 11-15 G 的单位阶跃响应

【例 11-18】 考虑一个高炮的伺服系统，其开环传递函数为

$$G_0(s) = \frac{5}{s(0.12s+1)(0.65s+1)}$$

(1) 在不加校正环节的情况下计算其幅值裕量和相角裕量。

(2) 在前向通道中加入校正环节 $R(s) = \frac{0.5s+1}{0.08s+1}$ 时求幅值裕量和相角裕量。

(3) 比较不加校正环节时的单位阶跃响应和加校正环节时的阶跃响应。假定反馈系数为 1。

解：

```
≫G0 = tf([5],conv([0.12 1 0],[0.65 1]))  %设置开环传递函数
Transfer function:
              5
---------------------------
0.078 s^3 + 0.77 s^2 + s
≫margin(G0)                  %绘制博德图及计算稳定裕量，如图 11-16 所示
≫[Gm,Pm] = margin(G0)        %幅值稳定裕量
Gm =                         %这里 Gm 与图 11-16 上标示的是不同的，前者为
                               增益裕量，后者为增益裕量的分贝值，实际上，
                               20lg1.9744 = 5.91dB
    1.9744
Pm =                         %相角稳定裕量
    14.8173
```

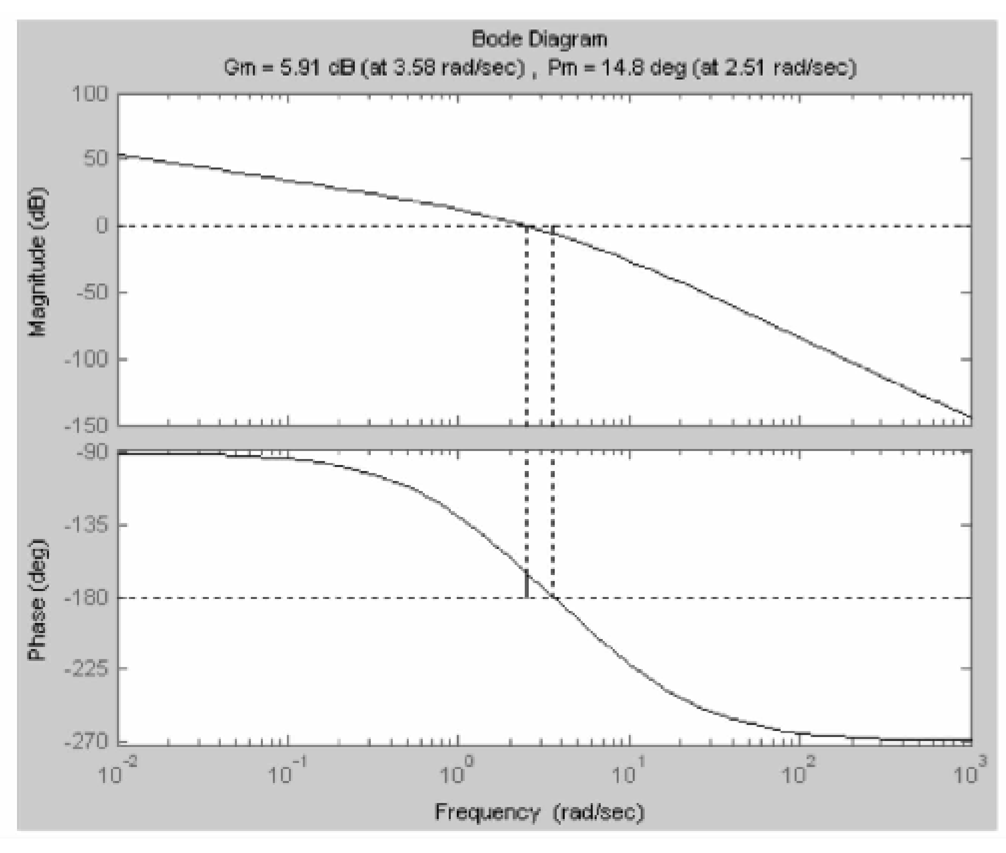


图 11-16 未加校正时，开环系统博德图

```
≫R = tf([0.5 1],[0.08 1])    %校正环节传递函数
```

Transfer function:
0.5 s + 1

0.08 s + 1
≫G1 = series(G0,R) %校正后，前向通道传递函数
Transfer function:
2.5 s + 5

0.00624 s^4 + 0.1396 s^3 + 0.85 s^2 + s
≫figure %重建图形窗口
≫margin(G1) %校正后的博德图，如图 11-17 所示

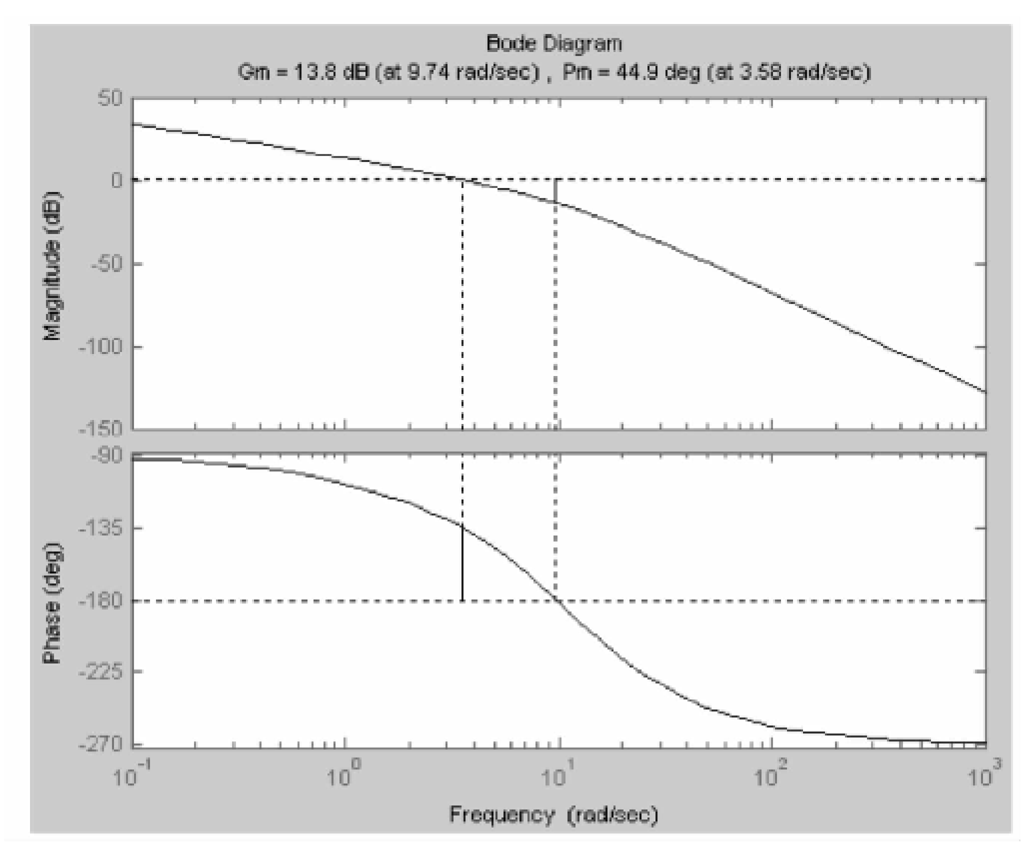


图 11-17 加校正后的博德图

≫[Gm1,Pm1] = margin(G1)
Gm1 = %校正后的幅值裕量
4.8966
Pm1 = %校正后的相角裕量
44.8676
≫G1 = series(G0,R)
≫G0c = feedback(G0,1) %未加校正的闭环传递函数
Transfer function:
5

```
-----------------------------------
0.078 s^3 + 0.77 s^2 + s + 5
>>step(G0c)                          %未加校正时的单位阶跃响应，如图 11-18 所示
>>hold on                            %图形保持
>>G1c = feedback(G1,1)               %加校正环节后的闭环传递函数
Transfer function:
                  2.5 s + 5
----------------------------------------------------------------
0.00624 s^4 + 0.1396 s^3 + 0.85 s^2 + 3.5 s + 5
>>step(G1c)                          %加校正环节后的单位阶跃响应，如图 11-18 所示
>>axis([0,10,0,1.7])                 %设置坐标
>>grid on                            %添加坐标格栅线
```

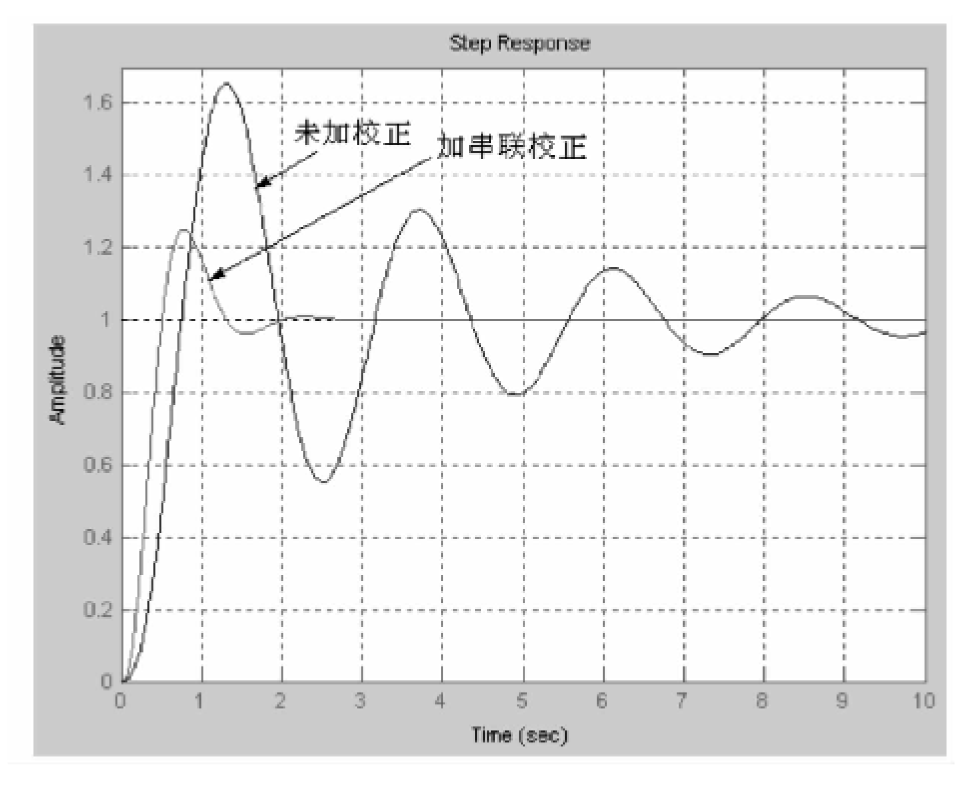


图 11-18　加串联校正与不加串联校正的阶跃响应的比较

11.7　任意输入作用下，控制系统的时间响应

在分析 LTI 系统时，通常考虑的输入作用是脉冲输入或单位阶跃输入。但有时需要正弦输入，方波输入或斜坡输入。正弦波输入常用于对控制系统的频率特性测试，方波输入则用于分析控制系统周期性工作的场合，斜坡输入则用于分析系统的跟随特性。

MATLAB 设置了线性系统输入仿真函数 lsim 来实现这一功能。lsim 函数定义为在任意输入作用下，LTI 系统输出响应。它的书写格式为

```
lsim(sys,u,t)
lsim(sys,u,t,x0)
```

lsim(sys, u, t, x0, 'zoh') or lsim(sys, u, t, x0, 'foh')

[y, t, x] = lsim(sys, u, t, x0)

式中，sys 为系统模型，t 为时间采样向量

t = 0：dt：Tfinal

dt 为时间间隔，Tfinal 为终止时间。矩阵 u 必须与时间向量有相同的行数，每一行 u(i, :) 指定输入值对应于采样时间 t(i)。如果控制系统用状态空间表示，则必须输入初始状态向量 x0。"zoh" 或 "foh" 用来表示采样间隔期间采用零阶保持或线性插值。在默认的情况下，lsim 选择插入方法，自动平滑信号 u。

当调用函数 lsim 没有左侧变量表达式时，则绘制控制系统的输出响应在屏幕上。与 lsim 相配合的函数为信号发生器函数 gensig，它用来产生正弦波、方法或脉冲波。它的书写格式为

[u, t] = gensig('type', tau, Tf, Ts)

式中，"type" 为信号类型，设定正弦用 "sin"；设定方波用 "square"；设定脉冲用 "pulse"。式中 tau 为信号周期，Tf 观察时间，Ts 为采样周期。作者对程序 gensig.m 稍加改动，程序名改为 gensigad.m 使后者程序能绘制锯齿波。

【例 11-19】 使用信号发生器函数 gensig 绘制正弦波、方波、锯齿波和脉冲波。Tu = 2；Tf = 10；Ts = 0.01。

解：

```
≫subplot(2,1,1)                          %设置子图 2 行 1 列第一图
≫[u,t] = gensig('sin',2,10,0.01);        %计算正弦信号数据
≫plot(t,u),axis([0,10,-1.2,1.2])         %绘制正弦信号图
≫title('正弦波')                          %输入标题，如图 11-19 所示
```

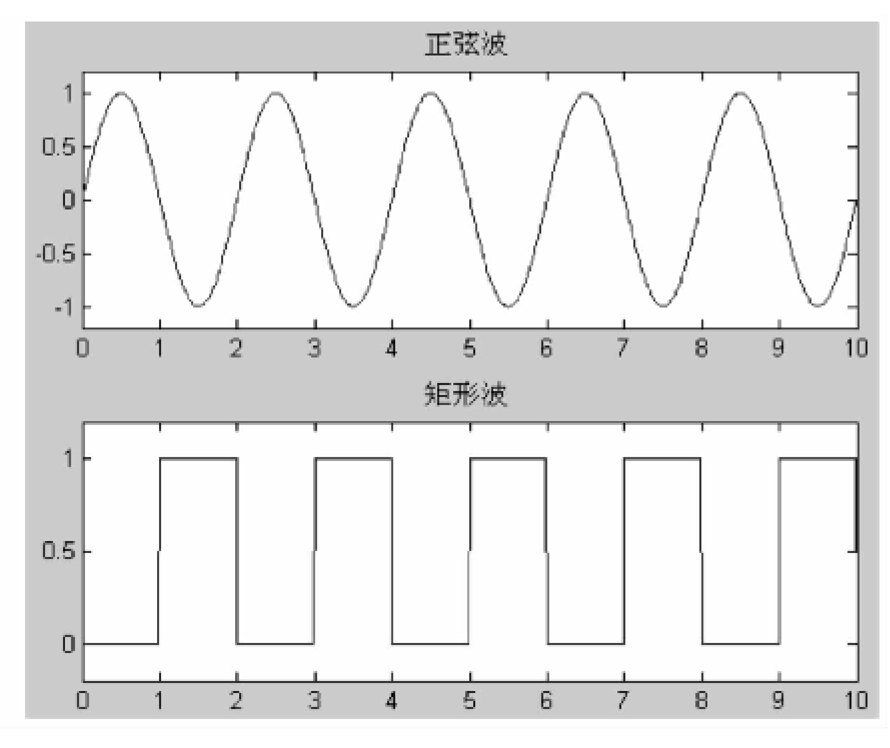


图 11-19　正弦波，矩形波信号图

```
≫subplot(2,1,2)                          %设置子图 2 行 1 列第二图
≫[u,t] = gensig('square',2,10,0.01);     %计算矩形信号数据
```

```
>>plot(t,u),axis([0,10,-0.2,1.2])          %计算矩形信号数据
>>[u,t]=gensig('sin',2,10,0.01);           %绘制矩形信号图
>>title('矩形波')                           %输入标题
>>figure                                    %创建图形窗口
>>subplot(2,1,1)                            %设置子图 2 行 1 列第一图
>>[u,t]=gensigad('pulse',2,10,0.01);       %计算锯齿波信号数据
>>plot(t,u),axis([0,10,-1.2,1.2])          %绘制锯齿波信号图
>>title('锯齿波')                           %输入标题，如图 11-20 所示
>>subplot(2,1,2)                            %设置子图 2 行 1 列第二图
>>[u,t]=gensig('pulse',2,10,0.01);         %计算脉冲信号数据
>>plot(t,u),axis([0,10,-0.2,1.2])          %绘制脉冲信号图
>>title('脉冲波')                           %设置标题
```

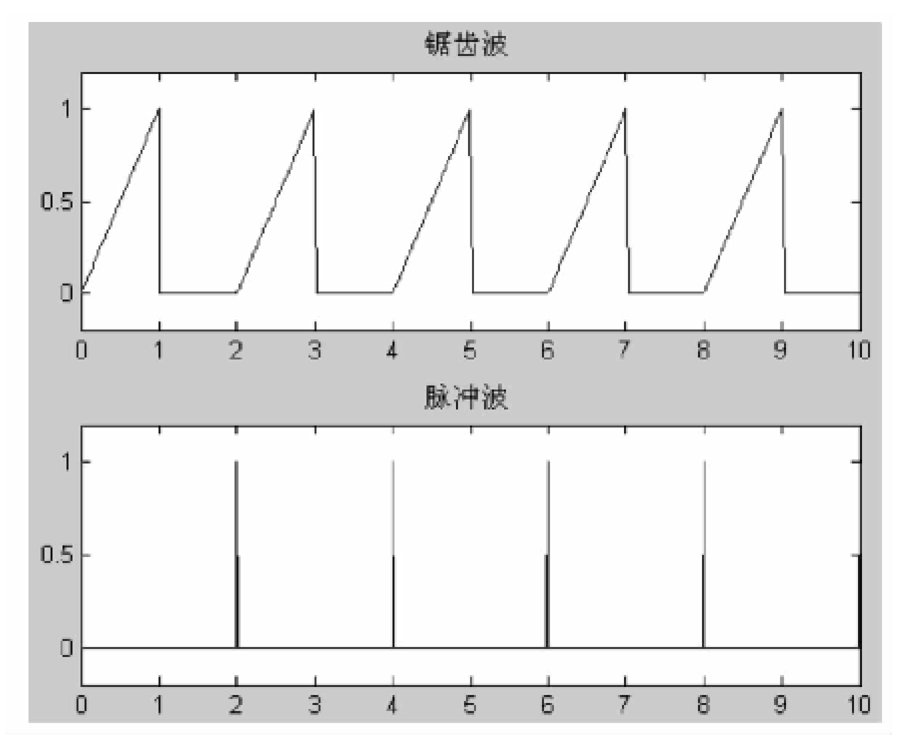


图 11-20　锯齿波，脉冲波的信号图

【例 11-20】　已知传递函数 $G=1/(0.01s^2+0.12s+1)$，求周期性矩形输入时的系统响应和锯齿波输入时的系统响应。Tau = 2，Tf = 10，Ts = 0.01，u 的幅值为 1。

解：

```
>>G=tf(1,[0.01 0.12 1])                     %创建传递函数
Transfer function:
          1
--------------------
0.01 s^2+0.12 s+1
>>[u,t]=gensig('square',2,10,0.01);        %计算矩形周期性信号数据
>>lsim(G,u,t)                               %输出响应，如图 11-21 所示
>>figure                                    %创建图形窗口
```

```
≫[u,t]=gensigad('pulse',2,10,0.01);   %计算锯齿波信号
≫lsim(G,u,t)                           %输出响应,如图 11-22 所示
```

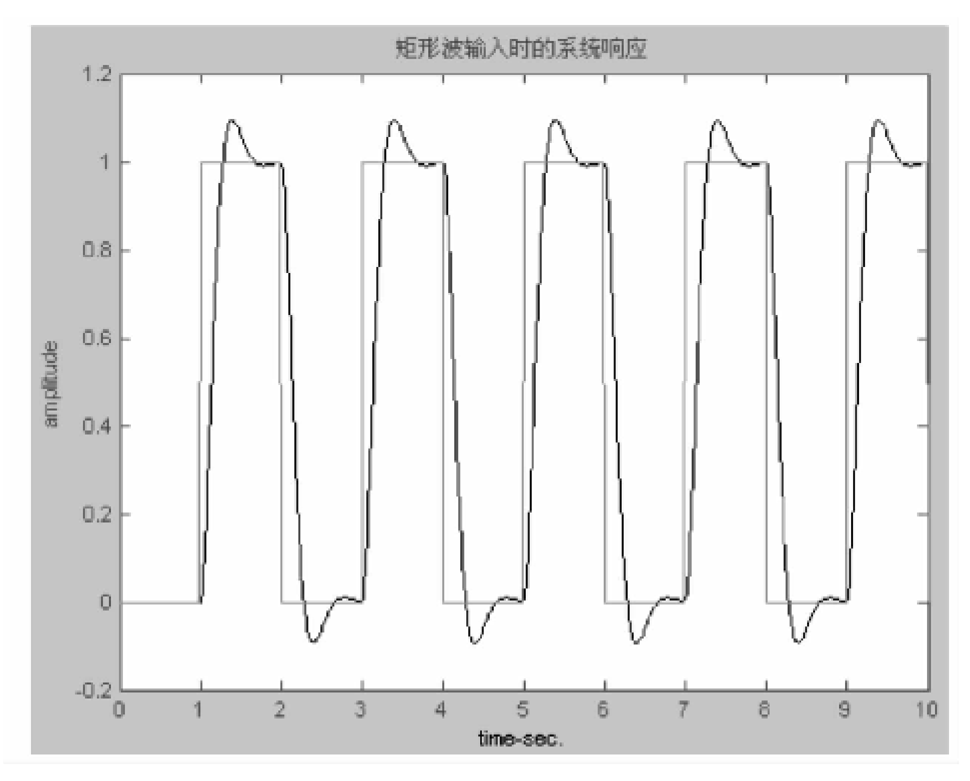


图 11-21　矩形波输入时的系统响应

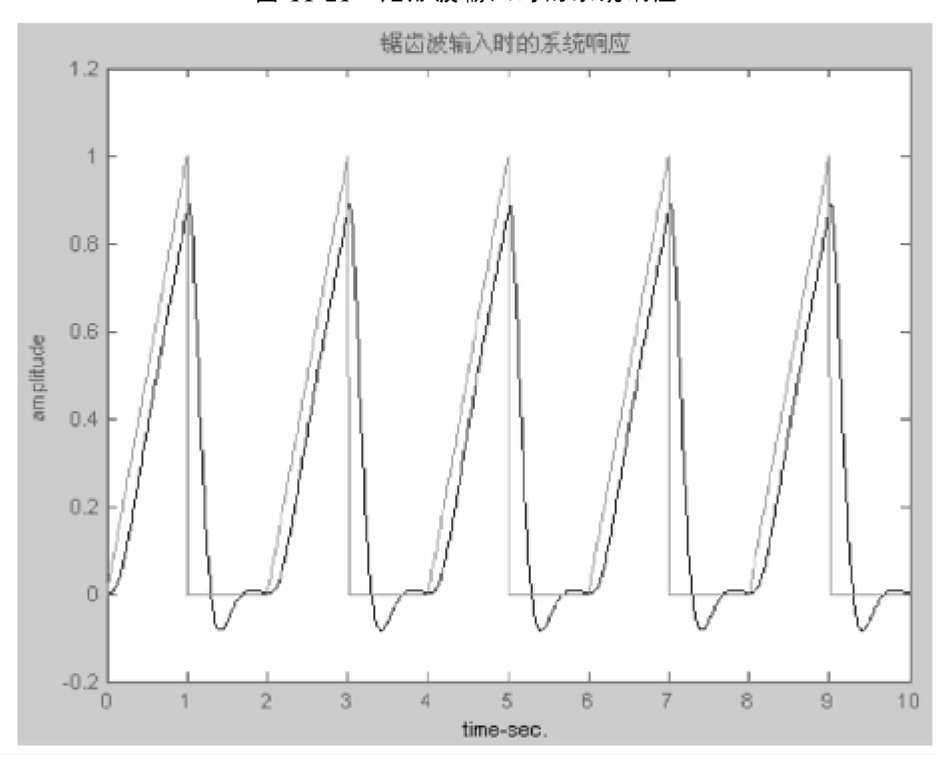


图 11-22　锯齿波输入时的系统响应

11.8　可控性与可观测性

考虑 LTI 系统的状态空间模型

$$\dot{x}(t)=Ax(t)+Bu(t),y(t)=Cx(t)$$

式中，$x(t)$是 n 维向量，$u(t)$为控制向量，$y(t)$为输出向量。如果在一个有限的时间间隔内施加一个控制向量，使得系统的初始状态$x(t_0)$转移到任一状态，则称该系统在 t_0 是可控的。

能控性的条件可以证明为

$$\text{rank}(\text{Co})=\text{n}$$

其中
$$\text{Co}=[B,AB,A^2B,\cdots,A^{n-1}B]$$

用 MATLAB 表示则为　　Co = ctrb(A,B)

式中，ctrb 为状态空间矩阵能控性函数。与能控性相类似的术语是能观测性。如果一个系统的状态$x(t_0)$都可以通过时间间隔 $t_0\leqslant t\leqslant t_1$ 内，由$y(t)$观测测定，则称该系统是完全可观测的。能观测性的充分必要条件可以证明为

$$\text{rank}(\text{Oc})=\text{n}$$

其中
$$\text{Oc}=[C',A'C',(A^2)'C',\cdots,(A^{n-1})'C']$$

用 MATLAB 表示则为　　Oc = obsv(A,C)

式中，obsv 为状态空间矩阵能观测性函数。

【例 11-21】　已知状态空间模型

$\boldsymbol{A}=\begin{vmatrix}-4 & 1 & 2\\ 1 & -5 & 3\\ 2 & 0 & -6\end{vmatrix}$，$\boldsymbol{B}=[1;\ 0.5;\ 2]$ $\boldsymbol{C}=[0\quad 1\quad 0]$，$\boldsymbol{D}=0$，试计算其可控性和可观测性。并列出它的传递函数。

解：

```
≫A = [-4 1 2;1 -5 3;2 0 -6]        %输入矩阵 A、B、C、D
A =
   -4     1     2
    1    -5     3
    2     0    -6
≫B = [1;0.5;2];
≫C = [0 1 0];D = [0];
≫Co = ctrb(A,B)                     %计算能控性矩阵
Co =
    1.0000     0.5000    -17.5000
    0.5000     4.5000    -52.0000
    2.0000   -10.0000     61.0000
≫rank(Co)                           %能控性矩阵的秩为 3 等于模型的阶次，故可控
ans =
     3
≫Oc = obsv(A,C)                     %计算能观测性矩阵
Oc =
    0     1     0
```

```
      1    -5     3
     -3    26   -31
>>rank(Oc)                          %能观测性矩阵的秩为3，等于模型的阶次，故可观测
ans =
      3
>>[num,den] = ss2tf(A,B,C,D)        %转换状态空间模型为传递函数的分子、分母多项式系数
num =
          0    0.5000    12.0000    50.0000
den =
     1.0000    15.0000    69.0000    88.0000
>>G = tf(num,den)                   %创建传递函数
Transfer function:
  0.5 s^2 + 12 s + 50
------------------------------
s^3 + 15 s^2 + 69 s + 88
```

【例 11-22】 已知状态空间模型

$$\begin{bmatrix} \dot{x}_1 \\ \dot{x}_2 \\ \dot{x}_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -6 & -11 & -13 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u, \boldsymbol{y} = \begin{bmatrix} 18 & 8 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

问该系统是完全可控和完全可观测吗？

解：

```
>>A = [0 1 0;0 0 1;-6 -11 -13];     %输入状态矩阵 A、B、C
>>B = [0;0;1];C = [18,8,1];
>>Co = ctrb(A,B)                    %计算可控性。由于状态矩阵为标准型，可以证明 ctrb (A, B) 是下三角矩阵，且对角元素全为 1
Co =
      0      0      1
      0      1    -13
      1    -13    158
>>rank(Co)                          %完全可控制
ans =
      3
>>Oc = obsv(A,C)                    %计算可观测性
Oc =
     18      8      1
     -6      7     -5
     30     49     72
```

```
>>rank(Oc)                              %完全可观测
    ans =
         3
```

11.9 极点配置

由于控制系统的过渡过程特性，取决于极点在左半复平面上的分布。所以极点的配置设计，就能使过渡过程在可接受的时间间隔内衰减完成。如果一个控制系统是可控的，且所有控制变量均可用于反馈，则可以实现将闭环系统的极点配置在 s 平面的任意位置。MATLAB 的极点配置函数为 place。

place　返回一个全状态反馈向量

$$K = place(A, B, p)$$

式中，p 为配置设计的极点分布向量。

【例 11-23】　已知 4 阶标准系统，其状态方程为

$$\boldsymbol{A} = \begin{bmatrix} -2 & 0 & 0 & 0 \\ 0 & -4 & 0 & 0 \\ 0 & 0 & -5 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \boldsymbol{B} = [1;1;1;10], \boldsymbol{C} = [1\ 0\ 0\ 0], \boldsymbol{D} = [0]$$

若极点配置在 $\boldsymbol{p} = [-1, -2.5, -4.5, -5.5]$，求状态反馈系数。

解：

```
>>A = diag([-2, -4, -5,0])             %输入状态矩阵
A =
    -2       0       0       0
     0      -4       0       0
     0       0      -5       0
     0       0       0       0
>>B = [1;1;2;10];
>>p = [-1, -2.5, -4.5, -5.5];           %配置极点向量
>>F = place(A,B,p)                      %计算状态反馈向量
F =
    0.3646     0.4219     0.0833     0.1547
>>A - B* F                              %配置极点后的状态矩阵
ans =
    -2.3646    -0.4219    -0.0833    -0.1547
    -0.3646    -4.4219    -0.0833    -0.1547
    -0.7292    -0.8437    -5.1667    -0.3094
    -3.6458    -4.2187    -0.8333    -1.5469
>>Ac = ans
Ac =
    -2.3646    -0.4219    -0.0833    -0.1547
```

```
    -0.3646    -4.4219    -0.0833    -0.1547
    -0.7292    -0.8437    -5.1667    -0.3094
    -3.6458    -4.2187    -0.8333    -1.5469
≫eig(Ac)                              %核对闭环状态矩阵的特征值是否与配置极点相同，结果是一致的
ans =
   -1.0000
   -2.5000
   -5.5000
   -4.5000
```

【例 11-24】 已知 I 型伺服系统，其开环传递函数为 $G(s)=1/[s(s+0.5)(s+1)]$希望闭环极点分布为$\boldsymbol{p}=[-2-\mathrm{j}2\ \ -2+\mathrm{j}2\ \ -5]$，当控制输入为单位阶跃函数，求输出响应。

解:

```
≫num = 1; den = conv([1,0.5,0],[1,1])  %输入 G(s)的分子、分母多项式 num，den
den =
    1.0000    1.5000    0.5000    0
≫G = tf(num,den)                       %开环传递函数 G(s)
Transfer function:
          1
---------------------
s^3 + 1.5 s^2 + 0.5 s
≫[A,B,C,D] = tf2ss(num,den)            %转换传递函数为状态空间模型
A =
    -1.5000    -0.5000    0
     1.0000          0    0
          0     1.0000    0
B =
    1
    0
    0
C =
    0    0    1
D =
    0
≫p = [-2-2*i, -2+2*i, -5];             %配置极点向量
≫F = place(A,B,p)                      %计算状态反馈向量
F =
    7.5000    27.5000    40.0000
≫A-B*F                                 %闭环后状态矩阵
```

```
ans =
    -9.0000    -28.0000    -40.0000
     1.0000           0           0
          0      1.0000           0
>>Ac = ans;                          %将闭环后状态矩阵赋予 Ac
>>B* F(3);                           %阶跃输入控制分量
>>Br = ans                           %赋予 Br
Br =
    40.0000
          0
          0
    >>[num1,den1] = ss2tf(Ac,Br,C,D) %转换闭环后状态模型为传递函数
num1 =
          0    -0.0000    -0.0000    40.0000
den1 =
    1.0000    9.0000    28.0000    40.0000
>>Gc = tf(num1,den1)
Transfer function:
              40
-----------------------------------
    s^3 + 9 s^2 + 28 s + 40
>>step(Gc),grid on                   %绘制阶跃响应，如图 11-23 所示
```

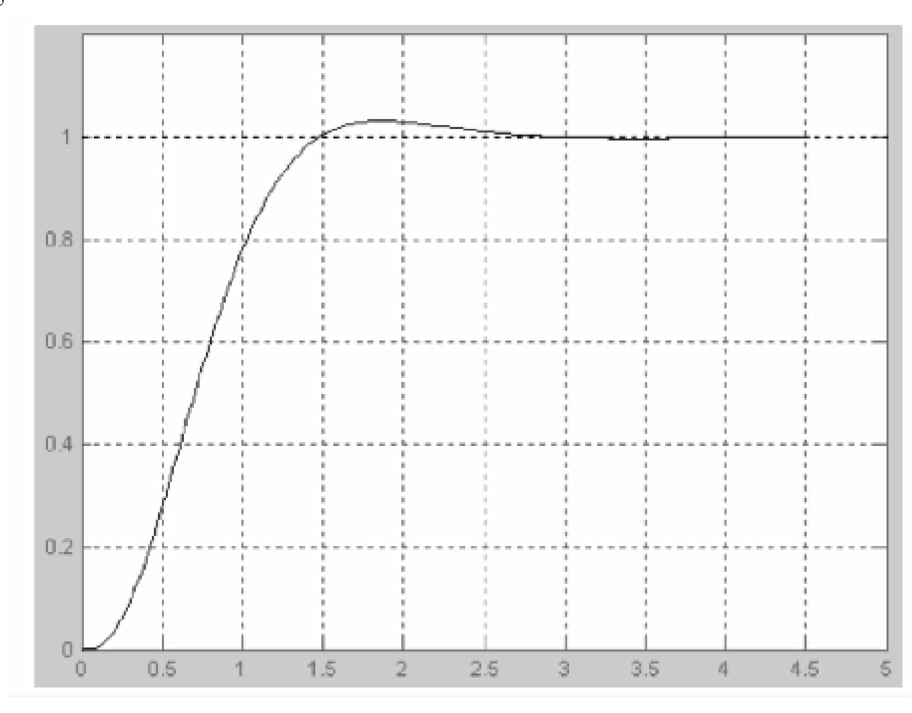


图 11-23　伺服系统的阶跃响应

```
>>axis([0,5,0,1.2])                  %设置坐标轴
```

第 11 章习题

11-1 考虑一个单输入 $u(t)$，单输出 $y(t)$ 系统，它是由下列微分方程式联系着的。

$$y'' + 4^{*} y' + 1 = u' + u$$

考虑初始条件为零，求系统的传递函数和零极点分布图。

11-2 已知系统的传递函数为

$$G(s) = \frac{s^2 + 3s + 1}{(s + 1.2)(s + 0.5)(s^2 + 1.7s + 1)}$$，求状态空间表示法。

11-3 已知前向传递函数为 $G(s) = \dfrac{50(0.05s + 1)}{(0.1s + 1)(0.015s + 1)(2s + 1)}$

反馈回路的传递函数为 $H = \dfrac{1}{0.04s + 1}$，求闭环传递函数。

11-4 已知系统的闭环传递函数为

$$G = \frac{1}{s^4 + 2.2s^3 + 3.4s^2 + 2.7s + 1}$$

用赫尔维茨行列式判别它的稳定性。绘制它的单位于阶跃响应。

11-5 考虑一个伺服系统，其开环传递函数为

$$G_0(s) = \frac{6}{s(0.08s + 1)(0.55s + 1)},$$

(1) 在不加校正环节的情况下计算其幅值裕量和相角裕量。

(2) 在前向通道中加入校正环节 $R(s) = \dfrac{0.45s + 1}{0.05s + 1}$ 时，求幅值裕量和相角裕量。

(3) 比较不加校正环节时的单位阶跃响应和加校正环节时的阶跃响应。假定反馈系数为 1。

第 12 章　MATLAB 在金融工作中的应用

在当前的社会中每个人或多或少总有些资产。这些资产可能是住房、土地、银行存款、股票、债券、期货、外汇、黄金、收藏品、矿藏、厂房、机器设备、甚至于企业的品牌等。资产持有者对于这些有形或无形的资产，总希望随着时间的推移，不断地扩大。但是在这过程中由于天灾、人祸、通货膨胀等因素会使你的资产贬值、挥发、衰退、仍至破产。这就需要有金融方面的知识来维护资产的保值，合理的安排资产结构以及规避金融风险是十分重要的。

金融工具，历来由银行存款、汇票、债券和股票组成。但是，从 20 世纪 70 年代以来。金融工具发生了很大的变化，除了大家熟悉的股票市场以外，还产生了期权市场。期权市场完全不同于人们熟悉的现金市场和商品市场。期权交易双方是签订一种合约，它赋予购买者一种权利，但并不附加义务，使得购买者有权在约定某一日期按照现在商定的价格购买或出售物品。在这种交易中买入方要向卖出方支付期权费。

MATLAB 有关金融方面有三个工具箱。即金融工具箱、金融时间序列工具箱和金融衍生工具箱。

MATLAB 及金融工具箱提供金融分析和金融工程的完整计算环境。在这工具箱内，有你需要的金融数据的数学和统计分析的一切资料，而且显示它的结果用上面介绍过的高品质图形。你能够利用它，询问、显示和回答复杂的金融问题。在传统的或电子表格编程中，你需要处理业务管理细目的所有类别、变量声明、数据类型、数据大小等，而 MATLAB 则为你做这一切。你只要为你所想的问题，写出数表达式就可以了。这里不需要切换工具，文件转换或重写应用程序。使用 MATLAB 和金融工具箱你可以实现以下任务。

(1) 计算和分析衍生工具和其他有价证券的定价、收益和灵敏度分析以及有价证券的组合投资。

(2) 执行有价证券协会（SIA）适用的固定收入的定价、收益和灵敏度分析。

(3) 设计与评估套期保值方案。

(4) 识别、检测和控制风险。

(5) 分析和计算现金流，包括投资回报率和贬值流。

(6) 分析与预报经济活动。

(7) 建立金融工具的结构，包括外汇交易工具。

(8) 教授与开展学术研究。

在金融衍生工具箱里，它扩充了金融工具箱在固定收入的衍生工具和有价证券的随机利率的范围内。这个工具箱提供了金融工具的分支，分析独立衍生的金融工具和组合投资。特别是它提供了必要的函数，为衍生金融工具的定价和灵敏度计算，为套期保值的计算，为分析结果的最优化服务。

金融时间序列工具箱，是一个用来汇集分析金融市场中的时间序列数据的工具。这工具箱包括了金融时间序列对象的构造器和几种分析对象的方法。金融工程工作方式具有时间系

列数据的，像证券的价格、每日收益的影响，都能够使用这工具箱，用有规律的向量和矩阵去处理数据，使数据更加直觉，然后用有规律性的向量或矩阵去管理数据。

MATLAB 有关金融方面的内容极为丰富。了解它的全部工具的用法，需要相应的金融方面的知识。MATLAB 的软件设计者期望用户具有下列学历和经验。

（1）分析师，数量分析师

（2）风险管理经理

（3）资金管理经理，资产管理经理

（4）经济学家

（5）金融工程师

（6）商人

（7）高等院校学生、教授和其他学术人员

限于作者的水平，本章只介绍按揭贷款的计算、投资组合的优化、工程投资回报率分析。股市蜡烛图线的绘制，风险的评估供读者参考。

12.1 住房贷款的等额本息还款法计算

当前在银行放贷项目中，住房按揭贷款占相当大的比重。这一贷款促进了房地产事业的迅速发展。它不但改善居民的居住条件，也促进了建筑行业，钢铁行业和相关行业的发展。与此同时创造了大量的就业机会。在这中间银行利率起了巨大的杠杆作用。过高的利率会使按揭购房减少，房价下跌。过低的利息，则使按揭贷款购房者数量增加，当房源供不应求时，受供求关系的影响，房价将上涨。

假设某购房者向银行贷款的金额为 P_0，银行的月利率为 a，贷款期限为 n 月，每月还款金额为 R_f，求还款金额 R_f。

这个问题可以用等比级数来求解，也可以用差分方程来求解。现在用前者来求解。

第一个月末贷款的本息和为

$$P(1) = P_0(1 + a) - R_f$$

第二个月末贷款的本息和为

$$P(2) = P(1)(1 + a) - R_f = P_0(1 + a)^2 - (1 + a)R_f - R_f$$

第三个月末贷款的本息和为

$$P(3) = P(2)(1 + a) - R_f = P_0(1 + a)^3 - (1 + a)^2 R_f - (1 + a)R_f - R_f$$

……

第 n 月末贷款的本息和为

$$\begin{aligned}P(n) &= P(n - 1)(1 + a) - R_f \\ &= P_0(1 + a)^n - [(1 + a)(n - 1) + (1 + a)^{(n-2)} + \cdots + (1 + a) + 1]R_f \\ &= P_0(1 + a)^n - R_f[(1 + a)^n - 1]/a\end{aligned}$$

考虑 n 月还贷完成，则 $P(n) = 0$，解 R_f 则得

$$\begin{aligned}R_f &= \frac{a(1 + a)^n}{(1 + a)^n - 1}P_0 \\ &= \frac{aP_0}{1 - (1 + a)^{-n}}\end{aligned}$$

【例 12-1】 银行贷款 100 万，月利率 0.465% 借款期限 10 年，问每月应还款金额。10 年内共计支付多少利息？

解:

借贷月数 $n = 10 \times 12 = 120$

每月还款金额为

```
>>Rf = 0.00465* (1 + 0.00465)^120* 100* 1e5/((1 + 0.00465)^120 - 1)
  Rf =
       1.0892e + 005
```

即 10892 元

支付利息

$$
\begin{aligned}
Interest &= Rf^{*} 120 - 100^{*} 1e005 \\
&= 3.0708e + 006 \\
&= 307080 \text{ 元}
\end{aligned}
$$

12.2 风险的防范与投资组合的优化

我们面临的商品市场是一个不确定的市场。厂商所面临的需求很可能是不确定的。一种新产品的出现，你所经营的老产品很可能就会没有市场。你的设备、成熟的技术将不再有用。有时同行业的竞争，你同行的产品可能在广告宣传上、质量上、价格上、售后服务上略胜于你，你的品牌有可能因此而淘汰出局。在农业生产上，农业收益的好坏在很大的程度上取决于气候，而气候的好坏也是不确定的。所以防范风险是每个人所关心的事。

作为金融学需要对风险下定义，并且可以用来计算和比较。风险是对期望收益的不能达到的概率。零风险，则意味着没有风险。粗看起来把资金存入银行是没有风险的（除了银行倒闭），但如果考虑物价上涨或通货膨胀因素，则仍是有风险的。因为银行到期的存单的实际价值已经低于你存入时的价值。

风险按其性质分有好多种：

(1) 技术风险。这是由于技术上的不成熟和施工中缺陷所造成的风险。例如火箭发射失败，是属于技术风险。

(2) 信用风险。这是由于交易双方的某一方违约所造成的风险。

(3) 市场风险。这是由于市场价格变动，销售变化所带来的风险。

(4) 经营风险。这包括法律风险，货物发送风险，欺诈、偷盗风险等。

(5) 决策风险。这是由于提供决策的信息不明朗，但又要做出决策所造成的风险。

关于风险的防范一般采用

1. 投资的多角化

即把投资不要集中在一个项目上，一旦发生不可避免的风险，会造成全军覆灭的损失。俗话说狡兔三窟，连野兔都会分散筑巢来躲避灾难。从概率分析也可得出分散投资可以降低风险。假设单项投资失败的风险为 0.5，若把资金分为 3 份去分别投资。这三个投资项目是互不相关的。则这三项投资全部失败的概率为 0.5/8 = 0.0625。因此风险被降低了。投资的多角化，就是要求把投资不要集中在一个目标上，而应该是分散的。但是如何分散，分散的比例如何安排，这里有个最优化问题。

2. 参与保险

风险对个人来讲，可能是毁灭性的。但对保险公司来讲是一个数值不大的概率。所以用万人保一人，就是保险公司的宗旨。一旦投保人发生风险，保险公司根据投保条款作部分的赔偿或全额的赔偿。当然，投保人需向保险公司支付保险金。保险金的金额应该相当于期望的损失。

3. 收集真实可靠有关投资的信息

当决策者对投资的环境信息不完全掌握，或不真实时，则投资的决策风险将很大。例如你应该了解银行利率的变化，上市公司的财务报表，股价的移动平均线，蜡烛图线，人气指数，行业发展动向等信息，来决策股票的买卖。

【例 12-2】 有一位投资者有 50 万资金，拟对 3 种股票进行 3 年的投资，根据市场分析和统计预测，这 3 家公司的期望收益，收益的方差和这 3 家公司间的协方差见表 12-1。

表 12-1 3 家公司的期望收益、收益方差、公司间的协方差

项目 / 公司	期望收益（%）	收益的方差	协方差（A，B）	协方差（B，C）	协方差（C，A）
A	60	180	35		
B	20	110		105	
C	50	150			－30

(1) 求在 3 年期望收益不低于 40% 的情况下，使投资收益的方差最小时的投资分配比例。

(2) 设置投资收益的标准差小于 10%，使期望收益最大时的投资分配比例。

在这里补充说明几个名词术语和计算方法。

(1) 期望收益。期望收益是对不确定收益事件的加权平均数，而这权数就是每种收益状态的出现的概率。例如你购买某个股票，买入价为 10 元，到年末涨到 15 元的概率为 0.1，涨至 12 元的概率为 0.2，持平的概率为 0.7，那末你这只股票的期望收益为

$$0.1(15-10)+0.2(12-10)+0.7(10-10)=0.9 \text{ 元}$$

写成一般形式为，对于随机变量 x_i 的数学期望为

$$E(x) = \sum_{i=1}^{n} x_i p_i$$

式中，p_i 为随机变量 x_i 的出现的概率。

(2) 方差。衡量随机变量的波动程度，它是随机变量 x_i 与数学期望的离差平方平均值来表示。

$$var = \sum_{i=1}^{n} (x_i - E(x))^2 p_i$$

MATLAB 中有计算方差的函数 var，若随机变量用向量 $\boldsymbol{X}$ 表示，则其方差即为 var($\boldsymbol{X}$)。

若已知随机变量 $\boldsymbol{X}$ = [12 9 9.3 7 7.2 7.5 14.2 13 8.5 8.7]，求方差。

```
>>var(X)
ans =
```

6.4293

(3) 标准差。标准差为方差的平方根sigma = sqrt(var(X))

(4) 协方差。协方差是描述两个随机变量相互关系的量。若两个随机变量为 X、Y，则它的协方差为

$$E\{[X - E(X)][Y - E(Y)]\}$$

在 MATLAB 中用函数cov(X,Y)表示协方差，若已知**X** = [11.5 11.5 10 10.5 10 11.2 9.5 9 8.8 8.5]，**Y** = [10 9.5 9 11 8.5 10 10.5 11.2 10.3 11.5]求协方差。

```
≫X = [11.5,11.5,10,10.5,10,11.2,9.5,9,8.8,8.5];    %随机变量 X
≫Y = [10,9.5,9,11,8.5,10,10.5,11.2,10.3,11.5];     %随机变量 Y
≫E = cov(X,Y)                       %显示协方差矩阵,其中 E(1,1) = var(X);
                                    %E(2,2) = var(Y);
                                    %E(1,2) = E{[X - E(X)]* [Y - E(Y)]}
                                    %E(2,1) = E{[X - E(Y)]* [Y - E(X)]}
    E =
      1.2339     -0.4872
     -0.4872      0.9228
≫var(X)                             %随机变量 X 的方差
    ans =
    1.2339
≫var(Y)                             %随机变量 Y 的方差
    ans =
    0.9228
```

解：

方案 (1) 分析。设投资分配比例，对 A 公司为$x(1)$，对 B 公司为$x(2)$，对 C 公司为$x(3)$，利用随机变量 x、y 和的方差计算公式

$$D(x,y) = D(x) + D(y) + 2\mathrm{cov}(x,y)$$

计算组合投资的方差为

$$Z = 180x(1)^2 + 110x(2)^2 + 150x(3)^2 + 70x(1)x(2) + 210x(2)x(3) - 60x(3)x(1)$$

根据最优化要求，组合投资的方差即为目标函数使之最小

$$\min Z$$

s.t. $x(1) + x(2) + x(3) = 1$ %投资总数为 1 的约束

$1.6x(1) + 1.2x(2) + 1.5x(3) \geqslant 1.4$ %期望收益大于 40%

$x(1)、x(2)、x(3) \geqslant 0$

接着编制目标函数文件，文件名为 varmin.m

```
function z = varmin(x)
z = 180* x(1)^2 + 110* x(2)^2 + 150* x(3)^2 + 70* x(1)* x(2) + 210* x(1)* x(3) - 60* x(2)* x
(3);
```

在命令窗口输入如下程序：

```
≫A = [-1.6, -1.2 - -1.5];b = -1.4;                 %输入不等式约束
```

```
≫Aeq = [1,1,1]; beq = 1                        %输入等式约束
≫lb = [0;0;0]; ub = [1;1;1]                    %输入下限与上限
≫x0 = [0;0;0]                                  %输入初值
≫options = optimset('fmincon');               %最优化设置成默认设置
≫options = optimset('largescale','off')       %不执行大规模算法
≫[x,fval] = fmincon(@varmin,x0,A,b,Aeq,beq,lb,ub,[],options)
                                               %调用条件约束最小化函数
Optimization terminated successfully:          %最优化成功
Magnitude of directional derivative in search direction  %在搜索方向的方向导数的幅值是小于
                                               2×目标函数的允差，而最大约束未违
                                               反，小于允许约束条件
  less than 2* options.TolFun and maximum constraint violation
  is less than options.TolCon
Active Constraints:                            %约束条件有效
      1
  x                                            %最优解
    0.3333
    0.3333
    0.3333
fval                                           %目标函数最小值
  73.3333
≫sqrt(fval)                                    %标准差为 8.5635%
  ans =
    8.5635
```

由以上分析说明，为了达到期望收益大于 40%，且收益的方差最小，则 A、B、C 三家公司的股票比例应各占 1/3。上述分析与计算是可靠的，但难度在于在实际操作中预测期望收益和方差估计。由于各个股票的收益率和方差经常在变动，所以在实际操作中需要随时调整投资比例，更换期望收益为负的股票。

方案 (2) 分析。目标函数改为期望收益，要求实现目标函数最大，目标函数为

$$\max Z_1 = 1.6x(1) + 1.2x(2) + 1.5x(3)$$

s.t. $x(1) + x(2) + x(3) = 1$ %投资总数为 1 的约束

$c = 180x(1)^2 + 110x(2)^2 + 150x(3)^2 + 70x(1)x(2) + 210x(2)x(3) - 60x(3)x(1) - 100 \leqslant 0$ %标准差小于 10% 的约束

编制目标函数的函数 M 文件，文件名为 earn.m

```
function z = earn(x)
z = -1.6*x(1) - 1.2*x(2) - 1.5*x(3);          %取负值是因为求目标函数最大
```

编制非线性约束的函数 M 文件，文件名为 varcon.m

```
function [c,ceq] = varcon(x)
c = 180*x(1)^2 + 110*x(2)^2 + 150*x(3)^2 + 70*x(1)*x(2) + 210*x(1)*x(3) - 60*x
```

```
(2)*x(3)-100;
    ceq=0;
```

在命令窗口输入如下程序：

```
>>options=optimset('maxfunevals',2000,'largescale','off');          %最优化设置
>>Aeq=[1,1,1];beq=1;                                                %等式约束
>>lb=[0;0;0];ub=[1;1;1]                                             %上、下限设置
>>x0=[0;0;0];                                                       %初值设置
>>[x,fval]=fmincon(@earn,x0,[],[],Aeq,beq,lb,ub,@varcon,options)
                                                                    %优化计算
Optimization terminated successfully:                               %最优化成功
Magnitude of directional derivative in search direction
  less than 2*options.TolFun and maximum constraint violation
  is less than options.TolCon
Active Constraints:
     1
x =                                                                 %最优解
    0.4478
    0.1607
    0.3914
fval =                                                              %目标函数最大值
  -1.4966
```

由以上分析可知，方案（2）的期望收益高于方案（1）的方差最小的分析结果。因为方案（2）中股票 A 的比例被加大了，股票 B 则缩小了。但是期望收益的方差也加大了。在这两种分析结果中究竟应选取那一种投资比例更好呢？这完全取决于投资者的风险偏好。经济学家把风险偏好分为三类人，即风险爱好者（risk love）风险中性（risk neutral）和风险规避者（risk evader)。其中大多数人则是风险规避者。对于本例的方案（1)、(2）的分析，风险中性和风险规避者愿意选取方案（1)，而风险爱好者则愿意选取方案（2)。

12.3 资金流的计算

大家知道现金具有时间价值，当前的现金 10000 元，如果月利率是 0.465%，那么 10 年后的面值为 $10000\times(1+0.00465)^{120}=17449$ 元。所以 MATLAB 为了现金流计算的方便，建立了一系列现金流计算函数，如年金、养老金计算，折旧计算、分期付款计算、现值计算、未来值计算和支付计算、回报率计算和灵敏度计算等。常用的现金流函数见表 12-2。

表 12-2 常见的现金流函数

函数名	说明
Annurate	年金的利率计算
Annuterm	零存整取，存期数计算
Amortize	分期付款计算

（续）

函　数　名	说　　明
depfixdb	固定递减折旧计算
depgendb	通用递减折旧计算
deprdv	残值保持折旧计算
depsoyd	年份折旧计算
depstln	直线折旧计算
pvfix	现值用固定周期支付
pvvar	变动现金流的现值
fvdisc	债券折扣的未来值
fvfix	用固定周期支付的未来值
fvvar	变动现金流的未来值
payadv	借贷资金的周期偿还
payodd	支付贷款用最初带零头的周期
payper	借贷资金的分期偿还
payuni	用均衡支付等同于可变现金流
effrr	有效回报率计算
irr	内部的回报率计算
mirr	内部的回报率修改
nomrr	名义回报率计算
taxedrr	税后回报率
xirr	内部的非周期性现金回报率
cfconv	凸性的现金流
cfdur	持续期和修改期的现金流

现将常用的现金流函数书写格式和用法介绍如下：

12.3.1　年金利率的计算

年金利率的计算函数 annurate 的书写格式为

rate = annurate(NumPeriods, Payment, PresentValue, FutureValue, Due)

式中，NumPeriods 为偿还期数；Payment 为每次偿还金额；PresentValue 为贷款现值；FutureValue 为贷款未来值（选项）；Due 为期末偿还 Due = 0（默认），期初偿还 Due = 1（选项）。

【例 12-3】　向银行贷款 50000 元，分 5 年还清，每月偿还 957 元，问银行的月利率。

解：

NumPeriods = 5 × 12 = 60

Payment = 957

PresentValue = 50000

在命令窗口输入如下程序：

```
≫format long                              %设置 15 位于显示格式
≫rate = annurate(60,957,50000,0,0)        %将数字代入 annurate 计算公式得月利率为 0.465%
rate =
  0.00465340886884
```

12.3.2 零存整取，存期数的计算

零存整取，存期数计算函数 annutermr，它的书写格式为

NumPeriods = annuterm (Rate, Payment, PresentValue, FutureValue, Due)

式中，NumPeriod 为达到存款总数所需存款月数。Rate 为月利率，PresentValue 为期望存款总数，Payment 为每期存款，其他变量的定义与上一节相同。

【例 12-4】 每月存入 2000 元，拟用于购房，银行年利率为 2.1%，问存款总额达到 50 万，需多长时间。

```
Rate = 0.021/12 = 0.00175                           %推算月利率
Payment = 2000
PresentValue = 500000
```

把上述数字代入 annuterm 函数则得存款期数

```
≫NumPeriods = annuterm (0.00175,2000,0,500000,0)    %存款期数
NumPeriods =
  207.5560
≫207.556/12                                         %折算年数
ans =                                               %需 17 年又 3 个半月
    17.2963
```

12.3.3 购物分期付款的计算

购物分期付款在当前社会是比较普遍的。对商家来说可以扩大销售，快速回收资金。对消费者来说可以提前享受高档消费品，而消费品的费用可以慢慢地偿还。由分期付款所产生的利息支付，一般由买、卖双方共同承担。MATLAB 中用 amortize 函数来计算分期付款。它的书写格式如下：

[Priciple, Interest, Balance, Payment] = amortize (Rate, NumPeriods, PresentValue, FutureValue, Due)

式中，Priciple 为本金向量；Interest 为利息向量；Balance 为剩余向量；Payment 为分期付款金额；NumPeriods 为分期付款次数；PresentValue 为物品原价；其他变量定义与上述相同。

【例 12-5】 一台笔记本电脑原价 11000 元，分 12 个月分期付款，月利率为 0.4%，问买者每月要偿还多少，并列出每期的本金、利息和剩余的款项。

解：

把上述数据代入 amortize 函数得

[Priciple, Interest, Balance, Payment] = amortize (0.004, 12, 11000, 0, 0)

```
Priciple =
  Columns 1 through 11
  896.6744   900.2611   903.8622   907.4776   911.1075   914.7520   918.4110   922.0846
  925.7729   929.4760   933.1939
  Column 12
  936.9267
Interest =
  Columns 1 through 11
  44.0000   40.4133   36.8123   33.1968   29.5669   25.9225   22.2635   18.5898
14.9015   11.1984   7.4805
  Column 12
    3.7477
Balance =
  1.0e+004*
  Columns 1 through 11
    1.0103   0.9203   0.8299   0.7392   0.6481   0.5566   0.4647   0.3725   0.2800
0.1870   0.0937
  Column 12
    0.0000
Payment =
  940.6744
```

每月应偿还 940.7 元。

12.3.4 设备折旧的计算

每项设备都有使用寿命，当一项设备的维修费用大于设备的残值时，这项设备只能报废。在设备尚未报废时，计算设备的残值称为折旧计算。设备折旧计算是有用的，企业中计算产品成本时，需要把生产机械的折旧费用考虑进去。旧货市场更要考虑设备的折旧率。折旧计算根据递减率不同分为

固定折旧——使用函数 depfixdb;

通用折旧——使用函数 depgendb;

残值折旧——使用函数 deprdv;

年份折旧——使用函数 depsovd;

直线折旧——使用函数 depstln。

限于篇幅，下面只介绍通用折旧，对于有兴趣的读者，只要在命令窗口输入 doc Function name 或 help function name 就可以了解函数的内容和用法。通用折旧函数的书写格式为

Depreciation = depgendb(Cost, Salvage, Life, Factor)

式中，Cost 为资产原值；Salvage 为残值评估；Life 为报废年限；Factor 为衰减因数，通常 factor = 2，为双倍-衰减-剩余方法。

depgendb 返回一个行向量，它计算每一个周期（以年表示）的递减贬值。

【例 12-6】 有辆轿车原价 250000 元，它的报废年限为 6 年，它的残值为 20000 元，衰减因数 factor = 2，问使用一年后，它的现值是多少。并计算它的折旧进程，用向量表示。

解：

```
cost = 250000; salvage = 20000; life = 6; factor = 2
Depreciation =                                        %折旧向量
  1.0e+004*
    8.3333   5.5556   3.7037   2.4691   1.6461   1.2922
```

使用一年后的现值只有 250000 元 − 83333 元 = 166667 元。

12.3.5 等额本息还贷

等额本息还贷在 MATLAB 中称贷款的周期性偿还，函数名为 payper，它的书写格式为

Payment = payper(Rate, Numperiods, PresentValue, FutureValue, Due)

式中，PresentValue 为当前贷款总金额；FutureValue 为未来值，或周期后的目标值，缺省时为 0；Numperiods 为偿还周期数，即偿还月数；Rate 为当前利率；Due 期末偿还 Due = 0（默认），期初偿还 Due = 1（选项）。

【例 12-7】 贷款 1000000 元，10 年还清，年利率 5.6%，问每月应偿还金额。若第 10 年终止日一次性还贷 500000 元，则每月应偿还金额。

解：

（1）每月应偿还金额

```
>>payment1 = payper(0.056/12, 120, 1000000, 0, 0)
payment1 =
  1.0902e+004
```

每月还贷 10902 元

（2）若第 10 年终止日一次性还贷 500000 元，则每月应偿还金额。

此时 FutureValue = −500000 元

```
>>payment2 = payper(0.056/12, 120, 1000000, -500000, 0)
payment2 =
  7.7845e+003
```

每月还贷 7784.5 元

12.3.6 用固定周期支付的未来值

用固定周期支付的未来值函数 fvfix，相当于零存整取，它采用固定周期和固定存款来计算未来的存款金额。它的书写格式和计算公式如下：

futurevalue = fvfix(rate, numperiods, payment, presentval, due)

这公式返回一个由一系列等额支付（存款）的未来值。

式中，rate 这月利率；numperiods 为周期数（月数）；payment 为每月支付金额（每月存款数）；presentvalue 当前值（首次存款，选项）；due 期末支付 Due = 0（默认），期初支付 Due = 1（选项）。

【例 12-8】 有一个储蓄账户开始时，账户剩余为 1500 元，每月存入 200 元，存 10 年，

年利率为 9%，以复利计算。问到期后，该储户账上的金额。

解:

rate = 0.09/12 = 0.0075;　　　　　　　　　　%估计月利率

numperiods = 10 * 12 = 120;

payment = 200;

presentvalue = 1500;代入上式得

≫futurevalue = fvfix(0.0075,120,200,1500,0)

futurevalue =

  4.2380e + 004

到期后，储户账上的金额应为 42380 元。

12.3.7 用固定周期支付的当前值

用固定周期支付的当前值的函数 pvfix，它返回一当前值，这个值是由一系列等周期，等值支付形成的。pvfix 的书写格式和计算公式如下：

presentvalue = pvfix(Rate,NumPeriods,Payment,ExtraPayment,Due)

式中，presentvalue 为当前值；extrapayment 为在期末的额外支付；其他变量的定义与 12.3.6 节相同。

【例 12-9】 每月存款 1000 元，月利率为 0.32% 存期为 10 年，求现值。并用其他算法来核对计算结果。

解:

rate = 0.0032;

numperiods = 10 × 12 = 120;

payment = 1000;把数据代入，则当前值为

≫presentvalue = pvfix(0.0032,120,1000,0,0)

presentvalue =

  9.9516e + 004

因此 10 年的现值为 99516 元，小于存款累计数 120 × 1000 = 12000 元。

presentvalue 的未来值

  ≫fv = presentvalue * (1 + 0.0032)^120

fv =

  1.4601e + 005

用固定周期支付的未来值函数 fvfix 计算 10 年存款的未来值为

≫fv1 = fvfix(0.0032,120,1000,0,0)

fv1 =

  1.4601e + 005

与 fv 完全相同。核对结果表明，本例的现值计算是正确的。

12.4 工程投资的回报率分析

工程的投资与开发是一项复杂的系统工程。它首先需要拟定一份可行性报告。在其中

(1) 分析项目建立的必要性和发展前景。

(2) 工程的环境分析，包括必要的地质、气象；原料供应；销售和运输；人才的来源；配套行业的情况等。

(3) 资金规划。分析分期的资金需要量。资金的来源，这包括自筹资金和银行贷款。还贷计划。

(4) 建造规划，何时动工，何时建成，建造中的资金需求。

(5) 生产规划，生产设备的购置，人员的招聘，培训等。拟定试生产的正式生产的计划。生产许可的报批等。

(6) 销售规划，这包括广告策划，产品的试销、推广工作等。

在这诸多工作中，投资回报率是十分重要的一环。如果一个工程，投资下去没有投资回报。那么这工程就没有商业投资价值。但是有的工程虽然没有商业投资价值，却有社会价值。如环保工程，教育培训工程，疾病防治工程等就不能以盈利为目的。

工程投资有这样的形式，在建厂期间，分期向银行贷款，工厂建成后，进行试生产，试生产完成后进行正式生产，正式生产中取得的利润进行逐步还贷。在这种情况下由于借贷和还贷的金额不固定，时间也不固定，所以未来值的计算必需采用变动现金流未来值函数 fvvar。变动现金流未来值函数的书写格式和计算公式如下：

FutureVal = fvvar(Cashflow, Rate, IrrCFDates)

式中，Futureval 为变动现金流的未来值；Cashflow 为变动现金流向量；Rate 为利率，假定借款与存款的利率量相同的；IrrCFDates 为不规则现金流的存在日期，由日期字符串列向量表示。

【例 12-10】 某人初期投资 50000 元于超市连锁店。年利率为 5.58%，第 1～5 年的收入为 6000、7500、11000、15000 和 19000 元，求这些规则现金流的未来值。

解：

```
cashflow = [-50000,6000,7500,11000,15000,19000];
rate = 0.0558;
```

未来值为

```
>>futureval = fvvar([-50000,6000,7500,11000,15000,19000],0.0558)
futureval =
  -2.2149e+003
```

未来值为 -2214.9 元，即原先投资尚未收回。

【例 12-11】 同上例但到款日期如下：

	到款日期
投资 50000 元	01/12/1999
收入 6000 元	02/24/2000
7500 元	03/03/2001
11000 元	04/08/2002
15000 元	05/15/2003
19000 元	07/01/2004

解：

```
%设置不规则现金流的存在日期
```

```
≫irrcfdates = ['01/12/1999';'02/24/2000';'03/03/2001';'04/08/2002';'05/15/2003';'07/01/2004'];
    %计算未来值
≫fv = fvvar([-50000,6000,7500,11000,15000,19000],0.0558,irrcfdates)
    fv =
    -3.3330e+003
```

未来值为-3333元，因推迟到款，所以比上例未来值更小。

【例12-12】 某工程总投资2亿，向银行分4次贷款，每年贷款5000万，年利率为5.58%，建成后试生产两年，试生产期间年利润1500万，第6年开始正式生产，年利润为4000万，工厂将利润用于还贷，问何时能回收投资，并问20年时未来值。

解：

第1~20年资金流向量为（以百万为单位，负值表示投资，正值表示利润）

cashflow = [-50, -50, -50, -50,15,15,40,40,40,40,40,40,40,40,40,40,40,40,40,40];

rate = 0.0558/12 = 0.00465

编制1~20年的未来值向量,程序名为ratertn.m

```
%This program for calculate the return of investment
cashfl = [-5, -5, -5, -5,1.5,1.5,4,4,4,4,4,4,4,4,4,4,4,4,4,4];   %现金流向量（单位为千万元）
rate = 0.0558;                                                   %估算月利率
for i = 1: 20                                                    %计算1~20年中，每一年的现金流的未来值
    cashflow = cashfl (1: i);
    fv (i) = fvvar (cashflow, rate);
end
fv                                                               %显示未来值向量
plot (fv), grid on                                               %绘制投资回报的线性图
title ('return of investment')                                   %线性图的标题
xlabel ('year')                                                  %横坐标的标签
ylabel ('ten mega yuan')                                         %纵坐标的标签，程序结束
```

运行程序如下：

```
≫ratertn
fv =                                                             %显示未来值向量
  Columns 1 through 11
    -5.0000   -10.2790   -15.8526   -21.7371   -21.4501   -21.1470   -18.3270
  -15.3496   -12.2061   -8.8872   5.3832
    Columns 12 through 20
    -1.6835    2.2225    6.3465    10.7007    15.2978    20.1514    25.2758    30.6862
36.3985
```

由投资回报的线性图（见图 12-1）可知，该工程预期在第 12 年初回收投资。并在第 20 年创造近 3.7 亿元利润。

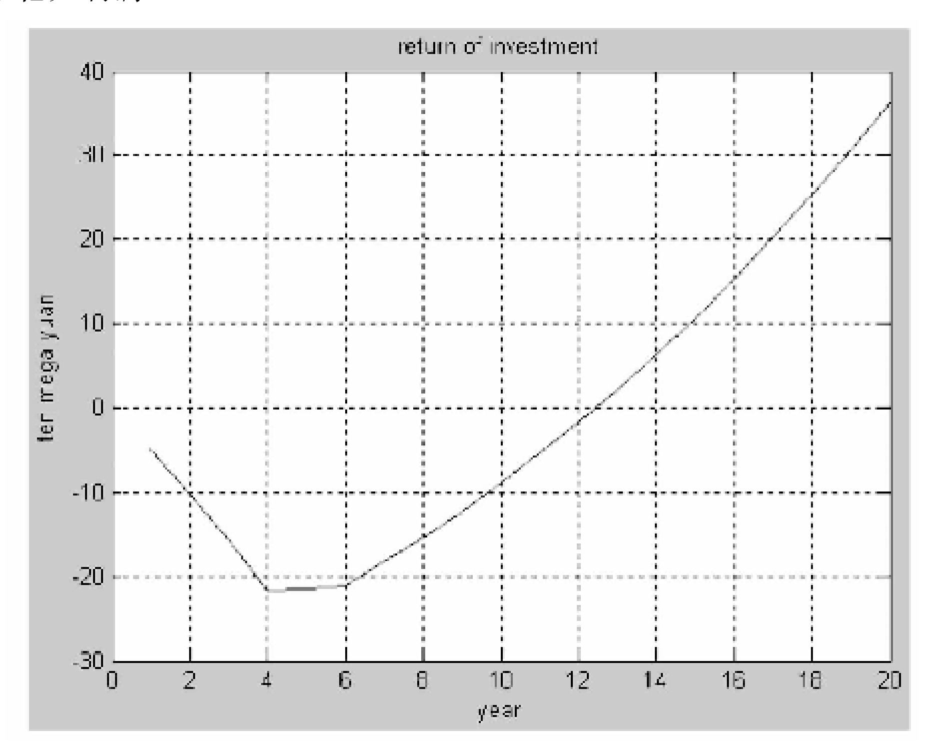


图 12-1 投资回报线性图

12.5 股市的蜡烛图线绘制

蜡烛图线又称 K 线图，它是用来表示股市变化的有用工具。它比移动平均线包含了更多的信息。MATLAB 中金融工具箱的绘图函数 candle，就是蜡烛图线的绘制工具。它的书写格式为

candle(Height, Low, Close, Open, Color)

式中，Height 为证券的最高价，以列向量表示；Low 为证券的最低价，以列向量表示；Close 为证券的收盘价，以列向量表示；Open 为证券的开盘价，以列向量表示；Color 为蜡烛体的颜色设置，用字符串表示。它支持默认设置，默认设置的颜色，取决于背景颜色。详见 ColorSpec。

假如收盘价大于开盘价，则蜡烛图线的蜡烛体是空心的。假如开盘价大于收盘价，则蜡烛图线的蜡烛体是涂色的。

上述蜡烛图的绘制是通过输入数据向量来实现，实际上还可以通过下载数据文件来实现。在这种情况下书写格式为

candle(tsobj, color)

式中，tsobj 为金融时间序列对象，由数据文件组成。在这种情况下，只要下载股市数据即可。若读者熟悉股市规律，下载股市实时行情，通过 MATLAB 的滤波工具、分析工具、编制分析软件，来预测股市，确定最佳买入点和最佳卖出点是可行的。

【例 12-13】 已知 04/01/2005 到 04/15/2005，上海证交所的浦发银行（代码 600000）的股价见表 12-3，绘制蜡烛图线。

表 12-3　2005 年 4 月 1 日 ~ 2005 年 4 月 15 日上海证交所浦发银行的股价

Date	Height	Low	Close	Open
04/01/2005	6.92	7.40	6.83	7.24
04/04/2005	7.05	7.11	6.96	7.00
04/05/2005	7.00	7.05	6.96	7.02
04/06/2005	7.00	7.20	6.85	7.19
04/07/2005	7.20	7.74	7.16	7.40
04/08/2005	7.40	7.58	7.32	7.52
04/11/2005	7.53	7.71	7.42	7.49
04/12/2005	7.48	7.50	7.36	7.38
04/13/2005	7.40	7.69	7.40	7.48
04/14/2005	7.48	7.54	7.33	7.35
04/15/2005	7.28	7.30	7.14	7.26

解：

在命令窗口输入如下程序：

```
>>Open = [6.92;7.05;7;7;7.2;7.4;nan;nan;7.53;7.48;7.4;7.48;7.28];
>>Height = [7.4;7.11;7.05;7.2;7.74;7.58;nan;nan;7.71;7.5;7.69;7.54;7.3];
>>Low = [6.83;6.96;6.96;6.85;7.16;7.32;nan;nan;7.42;7.36;7.4;7.33;7.14];
>>Close = [7.24;7;7.02;7.19;7.4;7.52;nan;nan;7.49;7.38;7.48;7.35;7.26];
>>A = [Open,Height,Low,Close]                %核对数据,NaN 为非交易日，故无数据
A =
    6.9200    7.4000    6.8300    7.2400
    7.0500    7.1100    6.9600    7.0000
    7.0000    7.0500    6.9600    7.0200
    7.0000    7.2000    6.8500    7.1900
    7.2000    7.7400    7.1600    7.4000
    7.4000    7.5800    7.3200    7.5200
       NaN       NaN       NaN       NaN
       NaN       NaN       NaN       NaN
    7.5300    7.7100    7.4200    7.4900
    7.4800    7.5000    7.3600    7.3800
    7.4000    7.6900    7.4000    7.4800
    7.4800    7.5400    7.3300    7.3500
    7.2800    7.3000    7.1400    7.2600
>>candle(Height,Low,Close,Open)              %绘制蜡烛图,如图 12-2 所示
```

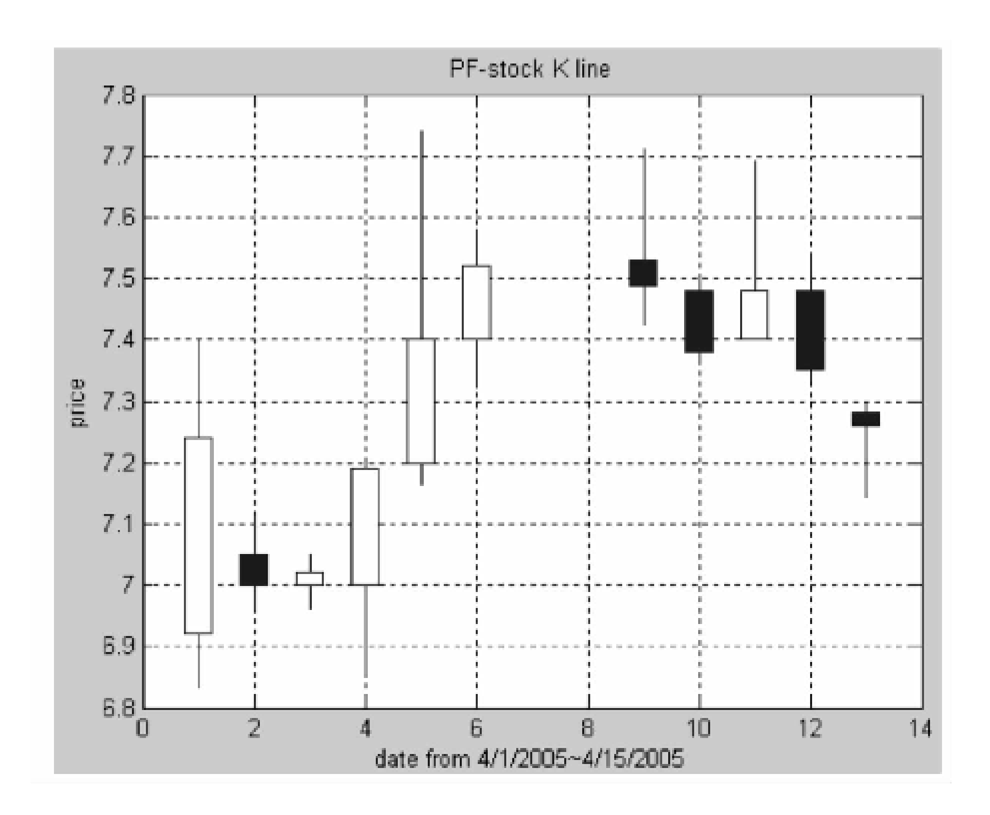


图 12-2 浦发银行的 K 线图（2005 年 4 月 1 日 ~ 2005 年 4 月 15 日）

第 12 章习题

12-1 某商店每天出售方便面数量的概率如下：

出售包数	50	100	150	200	250
概率	0.2	0.22	0.35	0.15	0.08

试用数学期望法确定每天最优进货量。

12-2 今有 A、B 两股票，一周内收盘价格波动如下表，求其方差值。

A	8.56	8.4	8.44	6.54	8.48
B	4.44	4.28	4.31	4.35	4.43

12-3 向银行贷款 50 万元，用于购房，月利率为 0.465%，借款期限为 5 年，问每月还款金额，使用函数 payper。

12-4 二手房原价 100 万元，报废年限为 20 年，残值为 30 万，问使用 5 年后的残值，采用直线折旧法（使用函数 depstln）。

12-5 某工程投资 1000 万元，分 5 次向银行贷款，每年 200 万元，年利率为 5.58%，建成后第 6 年投产，年利润为 160 万元。问投产几年后能回收投资。

12-6 某股票的开盘价、最高价、最低价和收盘价见下表：

Open	Max	Min	Close
4.38	4.52	4.37	4.44
4.41	4.44	4.27	4.28
4.28	4.35	4.26	4.31
4.32	4.39	4.28	4.35
4.38	4.45	4.34	4.43

试用 candle 函数绘制蜡烛图线。

参 考 文 献

1 Don M.Chance著.衍生金融工具与风险管理.郑磊译.北京：中信出版社，2004

2 干春晖主编.管理经济学.上海：立信会计出版社，2004

3 Frederick D.K.著.反馈控制问题.张彦斌译.西安：西安交通大学出版社，2001

4 徐树方等编.数值线性代数.北京：北京大学出版社，2000

5 韩大卫编著.管理运筹学.大连：大连理工大学出版社，2001

6 Katsuhiko Ogata著.现代控制工程.卢伯英等译.北京：电子工业出版社，2003